

# Moab HPC Suite – Basic Edition

RPM Installation Guide 8.0.0

July 2014



© 2014 Adaptive Computing Enterprises, Inc. All rights reserved.

Distribution of this document for commercial purposes in either hard or soft copy form is strictly prohibited without prior written consent from Adaptive Computing Enterprises, Inc.

Adaptive Computing, Cluster Resources, Moab, Moab Workload Manager, Moab Viewpoint, Moab Cluster Manager, Moab Cluster Suite, Moab Grid Scheduler, Moab Grid Suite, Moab Access Portal, and other Adaptive Computing products are either registered trademarks or trademarks of Adaptive Computing Enterprises, Inc. The Adaptive Computing logo and the Cluster Resources logo are trademarks of Adaptive Computing Enterprises, Inc. All other company and product names may be trademarks of their respective companies.

Adaptive Computing Enterprises, Inc.

1712 S. East Bay Blvd., Suite 300

Provo, UT 84606

+1 (801) 717-3700

[www.adaptivecomputing.com](http://www.adaptivecomputing.com)



*Scan to open online help*

# Contents

<b>Welcome</b> .....	<b>xxiii</b>
<b>Moab HPC Suite – Basic Edition release notes</b> .....	<b>1</b>
Moab HPC Suite – Basic Edition 8.0.0 release notes .....	1
New features .....	1
Differences .....	5
Installation and upgrade information .....	6
Known issues .....	7
Resolved issues .....	8
Key documentation changes .....	9
<b>Installation and Configuration</b> .....	<b>11</b>
Requirements .....	11
Manual installation .....	17
Installation .....	17
Preparing for installation .....	17
Installing TORQUE .....	23
Installing Moab Workload Manager .....	27
Installing Moab Web Services .....	31
Additional configuration .....	35
Configuring SSL in Tomcat .....	35
Setting up OpenLDAP on CentOS 6 .....	36
Moab Workload Manager configuration options .....	42
Trusting servers in Java .....	44
Upgrading .....	45
Upgrading MongoDB .....	45
Upgrading TORQUE .....	45
Upgrading Moab Workload Manager .....	47
Upgrading MWS .....	50
RPM installation .....	54
Installing Moab HPC Suite – Basic Edition .....	54
Configuration .....	59
Configuring TORQUE .....	59
Configuring Moab Workload Manager .....	61
Configuring Moab Web Services .....	62
Installing Insight .....	65
Installing Viewpoint .....	69
Additional configuration .....	71

Configuring SSL in Tomcat .....	71
Setting up OpenLDAP on CentOS 6 .....	71
Trusting servers in Java .....	77
Troubleshooting .....	78
Component documentation .....	86
<b>Moab Workload Manager .....</b>	<b>89</b>
Moab Workload Manager overview .....	89
Philosophy .....	89
Value of a Batch System .....	90
Philosophy and Goals .....	91
Workload .....	92
Scheduler Basics .....	94
Initial Moab Configuration .....	94
Layout of Scheduler Components .....	96
Scheduling Environment .....	98
Scheduling Dictionary .....	104
Scheduling Iterations and Job Flow .....	111
Configuring the Scheduler .....	114
Credential Overview .....	117
Job Attributes/Flags Overview .....	144
Scheduler Commands .....	151
Status Commands .....	153
Job Management Commands .....	154
Reservation Management Commands .....	155
Policy/Configuration Management Commands .....	156
End-user Commands .....	156
Commands .....	157
checkjob .....	157
checknode .....	169
mcredctl .....	174
mdiag .....	177
mdiag -a .....	180
mdiag -b .....	181
mdiag -c .....	181
mdiag -f .....	185
mdiag -g .....	187
mdiag -j .....	187
mdiag -n .....	188
mdiag -t .....	195
mdiag -p .....	195

mdiag -q .....	198
mdiag -r .....	199
mdiag -R .....	203
mdiag -S .....	208
mdiag -s .....	209
mdiag -T .....	209
mdiag -u .....	211
mjobctl .....	212
mnodectl .....	229
moab .....	235
mrmctl .....	236
mrsvctl .....	238
mschedctl .....	264
mshow .....	271
mshow -a .....	272
mshow -a .....	283
msub .....	286
Applying the msub submit filter .....	303
Submitting Jobs via msub in XML .....	305
mvectl (Moab Virtual Container Control) .....	309
mvmctl .....	314
showbf .....	318
showq .....	321
showhist.moab.pl .....	331
showres .....	336
showstart .....	342
showstate .....	345
showstats .....	347
showstats -f .....	359
TIMESPEC .....	362
Deprecated commands .....	362
canceljob .....	362
changeparam .....	363
diagnose .....	363
releasehold .....	364
releaseres .....	366
resetstats .....	366

runjob .....	367
sethold .....	368
setqos .....	369
setres .....	370
setspri .....	374
showconfig .....	375
Prioritizing Jobs and Allocating Resources .....	376
Job Prioritization .....	377
Priority Overview .....	377
Job Priority Factors .....	378
Fairshare Job Priority Example .....	388
Common Priority Usage .....	390
Prioritization Strategies .....	392
Manual Job Priority Adjustment .....	393
Node Allocation Policies .....	393
Node Access Policies .....	403
Node Availability Policies .....	404
Task Distribution Policies .....	411
Managing Fairness - Throttling Policies, Fairshare, and Allocation Management .....	411
Fairness Overview .....	411
Usage Limits/Throttling Policies .....	414
Fairshare .....	432
Sample FairShare Data File .....	445
Charging a Workflow .....	446
Controlling Resource Access - Reservations, Partitions, and QoS Facilities .....	450
Advance Reservations .....	450
Reservation Overview .....	450
Administrative Reservations .....	455
Standing Reservations .....	457
Reservation Policies .....	457
Configuring and Managing Reservations .....	460
Personal Reservations .....	491
Partitions .....	494
Quality of Service (QoS) Facilities .....	498
Optimizing Scheduling Behavior – Backfill and Node Sets .....	507
Optimization Overview .....	507
Backfill .....	508
Node Set Overview .....	514
Evaluating System Performance - Statistics, Profiling, Testing, and Simulation .....	521
Moab Performance Evaluation Overview .....	521
Accounting: Job and System Statistics .....	521
Testing New Versions and Configurations .....	523

Answering What If? Questions with the Simulator .....	524
General Job Administration .....	525
Job Holds .....	525
Job Priority Management .....	526
Suspend/Resume Handling .....	527
Checkpoint/Restart Facilities .....	528
Job Dependencies .....	528
Job Defaults and Per Job Limits .....	530
General Job Policies .....	531
Using a Local Queue .....	539
Job Deadlines .....	542
Job Arrays .....	545
General Node Administration .....	552
Node Location .....	553
Node Attributes .....	556
Node Specific Policies .....	566
Managing Shared Cluster Resources (Floating Resources) .....	567
Managing Node State .....	571
Managing Consumable Generic Resources .....	573
Enabling Generic Metrics .....	575
Enabling Generic Events .....	578
Resource Managers and Interfaces .....	584
Resource Manager Overview .....	585
Resource Manager Configuration .....	588
Resource Manager Extensions .....	618
PBS Resource Manager Extensions .....	647
Adding New Resource Manager Interfaces .....	649
Managing Resources Directly with the Native Interface .....	650
Utilizing Multiple Resource Managers .....	661
License Management .....	663
Resource Provisioning .....	665
Managing Networks .....	666
Intelligent Platform Management Interface .....	669
Resource Manager Translation .....	672
Troubleshooting and System Maintenance .....	673
Internal Diagnostics/Diagnosing System Behavior and Problems .....	673
Logging Overview .....	676
Object Messages .....	682
Notifying Administrators of Failures .....	684
Issues with Client Commands .....	685
Tracking System Failures .....	686
Problems with Individual Jobs .....	688
Diagnostic Scripts .....	689
Improving User Effectiveness .....	691

User Feedback Loops .....	691
User Level Statistics .....	692
Enhancing Wallclock Limit Estimates .....	693
Job Start Time Estimates .....	693
Providing Resource Availability Information .....	694
Collecting Performance Information on Individual Jobs .....	694
Cluster Analysis, Testing, and Simulation .....	694
Testing New Releases and Policies .....	695
Testing New Middleware .....	699
Configuring Resources for Simulation .....	701
Workload Event Format .....	702
Interactive Simulation Tutorial .....	712
Checking the Queue Status .....	712
Determining Why Jobs Are Not Running .....	713
Controlling Iterations .....	714
Managing Reservations Applying to the Queue .....	715
Verifying Fair Scheduling .....	721
Taking the System Down for Maintenance .....	722
Green computing .....	723
Green computing overview .....	723
How-to's .....	733
Deploying Adaptive Computing IPMI scripts .....	733
Choosing which nodes Moab powers on or off .....	735
Enabling green computing .....	736
Adjusting green pool size .....	740
Handling power-related events .....	740
Maximizing scheduling efficiency .....	741
Putting idle nodes in power-saving states .....	742
Troubleshooting green computing .....	742
Object triggers .....	745
About object triggers .....	745
How-to's .....	747
Creating a trigger .....	748
Creating VM triggers .....	751
Using a trigger to send email .....	752
Using a trigger to execute a script .....	754
Using a trigger to perform internal Moab actions .....	754
Requiring an object threshold for trigger execution .....	755
Enabling job triggers .....	755
Modifying a trigger .....	756
Viewing a trigger .....	757
Checkpointing a trigger .....	757

References .....	758
Job triggers .....	758
Node triggers .....	759
Reservation triggers .....	761
Resource manager triggers .....	762
Scheduler triggers .....	763
Threshold triggers .....	764
Trigger components .....	765
Trigger exit codes .....	773
Node maintenance example .....	773
Environment creation example .....	774
Trigger variables .....	775
About trigger variables .....	775
How-to's .....	776
Setting and receiving trigger variables .....	776
Externally injecting variables into job triggers .....	777
Exporting variables to parent objects .....	777
Requiring variables from generations of parent objects .....	778
Requesting name space variables .....	778
References .....	779
Dependency trigger components .....	779
Trigger variable comparison types .....	779
Internal variables .....	780
Miscellaneous .....	781
User Feedback Overview .....	781
Enabling High Availability Features .....	783
Malleable Jobs .....	785
Identity Managers .....	786
Generic System Jobs .....	790
Database Configuration .....	793
SQLite3 .....	793
Connecting to a MySQL Database with an ODBC Driver .....	794
Connecting to a PostgreSQL Database with an ODBC Driver .....	797
Connecting to an Oracle Database with an ODBC Driver .....	799
Installing the Oracle Instant Client .....	806
Migrating Your Database to Newer Versions of Moab .....	808
Importing Statistics from stats/DAY.* to the Moab Database .....	815
Accelerators .....	815
Scheduling GPUs .....	816
Using GPUs with NUMA .....	816
NVIDIA GPUs .....	817

GPU Metrics .....	820
Intel® Xeon Phi™ Coprocessor Configuration .....	822
Intel® Xeon Phi™ Co-processor Metrics .....	827
Preemption .....	828
About preemption .....	828
How-to's .....	830
Canceling jobs with preemption .....	830
Checkpointing jobs with preemption .....	833
Requeueing jobs with preemption .....	834
Suspending jobs with preemption .....	837
Using owner preemption .....	840
Using QoS preemption .....	844
References .....	845
Manual preemption commands .....	845
Preemption flags .....	846
PREEMTPOLICY types .....	848
Simple example of preemption .....	848
Testing and troubleshooting preemption .....	852
Job templates .....	853
About job templates .....	853
How-to's .....	854
Creating job templates .....	854
Viewing job templates .....	855
Applying templates based on job attributes .....	855
Requesting job templates directly .....	856
Creating workflows with job templates .....	857
References .....	858
Job template extension attributes .....	858
Job template matching attributes .....	870
Job template examples .....	870
Job template workflow examples .....	871
Moab Workload Manager for Grids .....	872
Grid Basics .....	873
Grid Configuration Basics .....	881
Centralized Grid Management (Master/Slave) .....	882
Hierarchal Grid Management .....	882
Localized Grid Management .....	884
Resource Control and Access .....	885
Workload Submission and Control .....	888
Reservations in the Grid .....	888
Grid Usage Policies .....	889
Grid Scheduling Policies .....	891
Grid Credential Management .....	893
Grid Data Management .....	895

Grid Security .....	900
Grid Diagnostics and Validation .....	900
Data staging .....	901
About data staging .....	901
How-to's .....	903
Configuring the SSH keys for the data staging transfer script .....	903
Configuring data staging .....	905
Staging data to or from a shared file system .....	907
Staging data to or from a shared file system in a grid .....	911
Configuring the \$CLUSTERHOST variable .....	915
Staging data to or from a compute node .....	916
Configuring data staging with advanced options .....	920
References .....	922
Sample user job script .....	922
Appendices .....	923
Appendix A: Moab Parameters .....	923
Appendix B: Multi-OS Provisioning .....	1079
Event Dictionary .....	1098
Appendix D: Adjusting Default Limits .....	1213
Appendix E: Security .....	1217
Appendix F: Initial Moab Testing .....	1225
Appendix G: Integrating Other Resources with Moab .....	1227
Compute Resource Managers .....	1228
Moab-TORQUE Integration Guide .....	1228
TORQUE/PBS Integration Guide - RM Access Control .....	1231
TORQUE/PBS Config - Default Queue Settings .....	1232
Moab-SLURM Integration Guide .....	1232
Installation Notes for Moab and TORQUE for Cray .....	1236
Provisioning Resource Managers .....	1252
Validating an xCAT Installation for Use with Moab .....	1252
Hardware Integration .....	1255
Moab-NUMA Integration Guide .....	1255
Appendix H: Interfacing with Moab (APIs) .....	1259
Appendix I: Considerations for Large Clusters .....	1263
Appendix J: Configuring Moab as a Service .....	1268
Appendix K: Migrating from 3.2 .....	1269
Appendix R: Node Allocation Plug-in Developer Kit .....	1272
Appendix S: Scalable Systems Software Specification .....	1278
Scalable Systems Software Job Object Specification .....	1278
Scalable Systems Software Resource Management and Accounting Protocol (SSSRMAP) Mes- sage Format .....	1314

Scalable Systems Software Node Object Specification .....	1338
Scalable Systems Software Resource Management and Accounting Protocol (SSSRMAP) Wire Protocol .....	1347
Appendix W: Moab Resource Manager Language Interface Overview .....	1363
W.1 Moab Resource Manager Language Data Format .....	1363
W.2 Managing Resources with SLURM .....	1372
W.3 Moab RM Language Socket Protocol Description .....	1383
SCHEDCFG flags .....	1389

## **Moab Web Services .....** 1395

Moab Web Services overview .....	1395
Setup .....	1395
Moab Web Services setup .....	1395
Configuring Moab Web Services .....	1395
Setting up MWS security .....	1410
Securing the connection with Moab .....	1411
Securing the connection with MongoDB .....	1411
Securing client connections to MWS .....	1412
Securing the LDAP connection .....	1416
Securing the connection with the message queue .....	1417
Version and build information .....	1418
Access control .....	1420
About access control .....	1420
Access control .....	1420
API documentation .....	1423
About the API .....	1423
RESTful web services .....	1423
Data format .....	1425
Global URL parameters .....	1425
Requesting specific API versions .....	1428
Responses and return codes .....	1429
Error messages .....	1432
Pre and post-processing hooks .....	1434
Authentication .....	1443
System events .....	1444
Resources .....	1446
Resources introduction .....	1446
Access control lists (ACLs) .....	1448
Accounting .....	1451
Accounting Accounts .....	1451
Accounting Allocations .....	1455
Accounting Charge rates .....	1459
Accounting Funds .....	1463
Accounting Liens .....	1473
Accounting Organizations .....	1477

Accounting Quotes .....	1480
Accounting Transactions .....	1483
Accounting Usage records .....	1488
Accounting Users .....	1502
Credentials .....	1506
Diagnostics .....	1521
Distinct .....	1526
Events .....	1528
Images .....	1536
Job arrays .....	1545
Jobs .....	1547
Job templates .....	1569
Metric types .....	1571
Nodes .....	1573
Notification conditions .....	1580
Notifications .....	1585
Permissions .....	1593
Plugins .....	1599
Plugin types .....	1607
Policies .....	1611
Fairshare .....	1618
Principals .....	1627
Priority .....	1633
Reports .....	1636
Reservations .....	1646
Resource types .....	1654
Roles .....	1655
Standing reservations .....	1661
Reporting framework .....	1665
Overview of reporting framework .....	1665
Example report (CPU Utilization) .....	1668
Plugins .....	1672
About Moab Web Services plugins .....	1672
Plugin overview .....	1673
Plugin introduction .....	1673
Lifecycle states .....	1675
Events .....	1676
Custom web services .....	1676
Utility services .....	1677
Data consolidation .....	1677
Routing .....	1679
Plugin developer's guide .....	1679
Requirements .....	1680
Dynamic methods .....	1681

Logging .....	1682
in messaging .....	1683
Configuration .....	1685
Configuration constraints .....	1686
Individual datastore .....	1694
Exposing web services .....	1696
Reporting state data .....	1698
Controlling lifecycle .....	1701
Accessing MWS REST resources .....	1702
Creating events and notifications .....	1704
Handling events .....	1711
Handling exceptions .....	1713
Managing SSL connections .....	1714
Utilizing services or custom "helper" classes .....	1715
Packaging plugins .....	1720
Example plugin types .....	1729
Moab Workload Manager resource manager integration .....	1730
Configuring Moab Workload Manager .....	1731
Resource manager queries .....	1735
Plugin type management .....	1737
Listing plugin types .....	1737
Displaying plugin types .....	1738
Plugin type documentation .....	1738
Add or update plugin types .....	1739
Plugin management .....	1742
Listing plugins .....	1742
Creating a plugin .....	1742
Displaying a plugin .....	1743
Modifying a plugin .....	1744
Deleting a plugin .....	1745
Monitoring and lifecycle controls .....	1745
Setting default plugin configuration .....	1747
Plugin services .....	1747
Job RM service .....	1748
Moab REST service .....	1748
Node RM service .....	1750
Plugin control service .....	1750
Plugin datastore service .....	1752
Plugin event service .....	1755
SSL service .....	1757
Storage RM service .....	1757
Virtual machine RM service .....	1757
Plugin types .....	1758
Power Management Plugin .....	1758

References .....	1763
Client code samples .....	1763
Javascript code samples .....	1764
PHP code samples .....	1765
Perl code samples .....	1769
Python code examples .....	1771
cURL samples .....	1771
Configuration .....	1772
Resource reference .....	1782
Resources reference .....	1782
Fields: Access Control Lists (ACLs) .....	1783
Accounting .....	1796
Fields: Accounts .....	1796
Fields: Allocations .....	1800
Fields: Charge Rates .....	1806
Fields: Fund Balances .....	1809
Fields: Fund Statement Summary .....	1819
Fields: Fund Statements .....	1835
Fields: Funds .....	1850
Fields: Liens .....	1862
Fields: Organizations .....	1868
Fields: Quotes .....	1871
Fields: Transactions .....	1878
Fields: Usage Records .....	1884
Fields: Users .....	1890
Fields: Credentials .....	1893
Fields: Events .....	1894
Fields: Images .....	1903
Fields: Job Arrays .....	1915
Fields: Jobs .....	2006
Fields: Job Templates .....	2107
Fields: Metric Types .....	2149
Fields: Nodes .....	2150
Fields: Notification Conditions .....	2191
Fields: Notifications .....	2195
Fields: Plugins .....	2197
Fields: Plugin Types .....	2204
Fields: Policies .....	2210
Fields: Principals .....	2249
Fields: Report Datapoints .....	2259

Fields: Reports .....	2261
Fields: Reservations .....	2270
Fields: Resource Types .....	2330
Fields: Roles .....	2331
Fields: Report Samples .....	2341
Fields: Standing Reservations .....	2343
Fields: User's Permissions .....	2428
<b>Moab Cluster Manager .....</b>	<b>2434</b>
MCM overview .....	2434
Getting Started .....	2435
Installation .....	2435
Connection wizard .....	2436
View summary .....	2439
Workload .....	2442
Jobs .....	2443
Reservations .....	2458
Reservation Calendar .....	2469
Triggers .....	2469
Resources .....	2471
Visual cluster .....	2471
Nodes .....	2472
Node calendar .....	2479
Partitions .....	2480
Scheduler settings .....	2482
Credentials .....	2486
All credentials .....	2486
Visual credentials .....	2491
Create/Modify a user profile .....	2491
Create/Modify a group profile .....	2495
Create/Modify an account profile .....	2498
Create/Modify a class profile .....	2501
Create/Modify a quality of service profile .....	2504
Policies .....	2508
Backfill .....	2508
Fairshare .....	2510
Job policies and settings .....	2512
Node policies .....	2515
Partition policies .....	2517
Priority .....	2517
Reservation policies .....	2522
User access .....	2523
Reporting .....	2524
Custom charts/graphs .....	2524

Event calendar .....	2531
Matrix statistics .....	2531
Processor usage .....	2533
Quick charts/graphs .....	2534
Reports .....	2534
Command line charts/graphs/reports .....	2537
Diagnostics .....	2538
Diagnostics support .....	2538
Miscellaneous .....	2538
Console .....	2538
Debugging and log levels .....	2539
MCM preferences .....	2540
<b>TORQUE Resource Manager .....</b>	<b>2541</b>
Introduction .....	2541
Overview .....	2543
TORQUE installation overview .....	2544
TORQUE architecture .....	2544
Installing TORQUE .....	2544
Compute nodes .....	2549
Enabling TORQUE as a service .....	2550
Initializing/Configuring TORQUE on the server (pbs_server) .....	2551
Specifying compute nodes .....	2553
Configuring TORQUE on compute nodes .....	2554
Configuring ports .....	2555
Configuring trqauthd for client commands .....	2556
Finalizing configurations .....	2558
Advanced configuration .....	2558
Customizing the install .....	2558
Server configuration .....	2566
Manual setup of initial server configuration .....	2571
Server node file configuration .....	2572
Basic node specification .....	2572
Specifying virtual processor count for a node .....	2572
Specifying GPU count for a node .....	2573
Specifying node features (node properties) .....	2573
Testing server configuration .....	2574
TORQUE on NUMA systems .....	2576
TORQUE NUMA configuration .....	2576
Building TORQUE with NUMA support .....	2576
TORQUE Multi-MOM .....	2580
Multi-MOM configuration .....	2580
Stopping pbs_mom in Multi-MOM mode .....	2582
Submitting and managing jobs .....	2582

Job submission .....	2583
Multiple job submission .....	2584
Managing multi-node jobs .....	2585
Requesting resources .....	2586
Requesting generic resources .....	2593
Requesting floating resources .....	2593
Requesting other resources .....	2594
Exported batch environment variables .....	2594
Enabling trusted submit hosts .....	2596
Example submit scripts .....	2596
Job files .....	2597
Monitoring jobs .....	2599
Canceling jobs .....	2599
Job preemption .....	2600
Keeping completed jobs .....	2600
Job checkpoint and restart .....	2601
Introduction to BLCR .....	2601
Configuration files and scripts .....	2602
Starting a checkpointable job .....	2609
Checkpointing a job .....	2610
Restarting a job .....	2610
Acceptance tests .....	2611
Job exit status .....	2611
Service jobs .....	2615
Submitting service jobs .....	2616
Submitting service jobs in MCM .....	2616
Managing service jobs .....	2616
Managing nodes .....	2617
Adding nodes .....	2617
Node properties .....	2618
Changing node state .....	2618
Changing node power states .....	2619
Host security .....	2621
Linux cpuset support .....	2623
Scheduling cores .....	2624
Geometry request configuration .....	2624
Geometry request usage .....	2625
Geometry request considerations .....	2625
Scheduling accelerator hardware .....	2625
Setting server policies .....	2626
Queue configuration .....	2626
Queue attributes .....	2626
Example queue configuration .....	2637
Setting a default queue .....	2638

Mapping a queue to subset of resources .....	2638
Creating a routing queue .....	2638
Server high availability .....	2640
Setting min_threads and max_threads .....	2652
Integrating schedulers for TORQUE .....	2653
Configuring data management .....	2653
SCP setup .....	2653
Generating SSH key on source host .....	2654
Copying public SSH key to each destination host .....	2654
Configuring the SSH daemon on each destination host .....	2655
Validating correct SSH configuration .....	2655
Enabling bi-directional SCP access .....	2655
Compiling TORQUE to support SCP .....	2656
Troubleshooting .....	2656
NFS and other networked filesystems .....	2656
File stage-in/stage-out .....	2657
MPI (Message Passing Interface) support .....	2658
MPICH .....	2658
Open MPI .....	2659
Resources .....	2660
Accounting records .....	2662
Job logging .....	2664
Job log location and name .....	2664
Enabling job logs .....	2664
Troubleshooting .....	2665
Host resolution .....	2666
Firewall configuration .....	2666
TORQUE log files .....	2666
Using "tracejob" to locate job failures .....	2668
Using GDB to locate job failures .....	2670
Other diagnostic options .....	2670
Stuck jobs .....	2671
Frequently asked questions (FAQ) .....	2672
Compute node health check .....	2677
Configuring MOMs to launch a health check .....	2678
Creating the health check script .....	2678
Adjusting node state based on the health check output .....	2679
Example health check script .....	2679
Debugging .....	2679
Appendices .....	2685
Appendix A: Commands overview .....	2686
momctl .....	2687
pbs_mom .....	2693
pbs_server .....	2699

pbs_track .....	2702
pbsdsh .....	2704
pbsnodes .....	2705
qalter .....	2709
qchkpt .....	2719
qdel .....	2720
qgpumode .....	2722
qgpureset .....	2723
qhold .....	2724
qmgr .....	2726
qmove .....	2729
qorder .....	2730
qrerun .....	2731
qrls .....	2733
qrun .....	2734
qsig .....	2735
qstat .....	2737
qsub .....	2744
qterm .....	2763
trqauthd .....	2765
Appendix B: Server parameters .....	2766
Appendix C: Node manager (MOM) configuration .....	2783
Parameters .....	2784
Node features and generic consumable resource specification .....	2801
Command-line arguments .....	2801
Appendix D: Diagnostics and error codes .....	2802
Appendix E: Considerations before upgrading .....	2810
Appendix F: Large cluster considerations .....	2811
Scalability guidelines .....	2812
End user command caching .....	2812
Moab and TORQUE configuration for large clusters .....	2814
Starting TORQUE in large environments .....	2815
Other considerations .....	2815
Appendix G: Prologue and epilogue scripts .....	2817
Script order of execution .....	2818
Script environment .....	2819
Per job prologue and epilogue scripts .....	2820
Prologue and epilogue scripts time out .....	2821
Prologue error processing .....	2821
Appendix H: Running multiple TORQUE servers and MOMs on the same node .....	2825
Appendix I: Security overview .....	2826
Appendix J: Job submission filter ("qsub wrapper") .....	2827
Appendix K: "torque.cfg" configuration file .....	2828
Appendix L: TORQUE Quick Start Guide .....	2833

Appendix M: BLCR acceptance tests ..... 2836

- Test environment ..... 2836
- Test 1 - Basic operation ..... 2837
- Test 2 - Persistence of checkpoint images ..... 2839
- Test 3 - Restart after checkpoint ..... 2840
- Test 4 - Multiple checkpoint/restart ..... 2841
- Test 5 - Periodic checkpoint ..... 2841
- Test 6 - Restart from previous image ..... 2842



# Welcome

Welcome to *Moab HPC Suite – Basic Edition 8.0.0*, which lets you...

This guide is intended for...

The following sections will help you quickly get started with Moab HPC Suite:

## About Moab HPC Suite

Gives an overview about Moab HPC Suite basics.

## Installing Moab HPC Suite

Instructs on how to install Moab HPC Suite.

## Configuring Moab HPC Suite

Explains how to configure and set up Moab HPC Suite.

## Common Moab HPC Suite tasks

Contains helpful task-based information for the most common Moab HPC Suite functions.

## Moab HPC Suite references

Contains useful examples, configurations, use cases and references relating to Moab HPC Suite use and setup.



# Moab HPC Suite – Basic Edition release notes

## Moab HPC Suite – Basic Edition 8.0.0 release notes

The release notes file contains the following sections:

- [New features on page 1](#)
- [Differences on page 5](#)
- [Installation and upgrade information on page 6](#)
- [Known issues on page 7](#)
- [Resolved issues on page 8](#)
- [Key documentation changes on page 9](#)

## New features

The following is a summary of key new features in Basic.

- [Moab Workload Manager on page 1](#)
- [Moab Web Services on page 4](#)
- [TORQUE Resource Manager on page 5](#)

### Moab Workload Manager

#### *Data staging*



The old method of data staging has been deprecated in Moab Workload Manager 8.0.0 and will be removed from the product in a future release.

Moab data staging has undergone a major redesign in 8.0.0. Consequently, Moab includes several new features that improve the data staging experience. These features include the following:

- Moab data staging system jobs, separately schedulable from the user job, that increase system performance and utilization by not reserving compute nodes during input or output data-staging

unless a compute node's local file system is involved.

- New `msub` options that are required to submit data staging jobs. At least one of `--stagein`, `--stageinfile`, `--stageout`, and/or `--stageoutfile` is required to stage data for your job.
  - `--stagein` and `--stageout` specify individual files and/or directories to stage in or out, respectively.
  - `--stageinfile` and `--stageoutfile` specify the path to an individual file that contain the paths of files and/or directories to stage in or out, respectively.
- `--stageinsize` and `--stageoutsize` options that give Moab an estimate of the size of the files or directories to stage in or out, respectively, so it can more accurately schedule the associated data staging jobs. For more information, see "[msub](#)".
- A customizable reference data staging job submit filter that sums the size of all input files and passes the sum to Moab. It uses the `--stageinsize` option so you do not have to estimate an input data size and Moab can more accurately estimate wallclock time when scheduling the input data staging system job. For more information, see "[Configuring data staging](#)".
- Customizable reference scripts that use `scp` or `rsync` to stage data (`/opt/moab/tools/data-staging/ds_move_scp` and `/opt/moab/tools/data-staging/ds_move_rsync`, respectively). The scripts work in an out-of-the-box environment, but you should modify one or both to work with your unique environment. You can use the reference scripts as a guide to create your own script that supports data staging with a different Linux file transfer utility or commercial file transfer utility, such as one from Aspera.
- The ability to set generic metrics on partitions (for more information, see "[Per-Partition Settings](#)").
  - The new `DATASTAGINGBANDWIDTH_MBITS_PER_SEC` metric, required for data staging, specifies the transfer rate of the partition in megabits per second. Moab uses it and the `--stageinsize` and/or `--stageoutsize` `msub` options submitted with the job to estimate the wallclock time for better scheduling of the input and output data staging system jobs.
  - The new metric can be dynamically updated by a resource manager or configured by an administrator.
- When you run `checkjob -v` on a data staging job, Moab returns the source and destination of data staging files and their total size. See "[checkjob](#)" for more information.
- Advanced configuration options allow you to rename the default template, support multiple file transfer script utilities in a grid on a per-partition basis, configure a notification email to be sent after a data staging job completes, add a non-default template via `msub`, and use `msub` to return all job IDs in the workflow at submission time. This is useful when you dynamically generate user workflows using scripts that must create dependencies on data-staging system jobs (when input or output data-staging has completed, for example), not the user job. For more information, see "[Configuring data staging with advanced options](#)".

The features associated with data staging in previous Moab releases have been deprecated in this release and will be removed in the next. For more information about data staging in Moab Workload Manager, see "[About data staging](#)".

### *CPU frequency control*

Moab can now ask the resource manager to change the CPU frequency on allocated nodes for submitted jobs. The request is made with the new [CPULOCK](#) resource manager extension. A user can specify a desired clock frequency in megahertz, a Linux power governor policy name, or an ACPI performance state (P-state) number.

The [mjobctl -m](#) command has been updated to allow modification of the requested CPU frequency on an already submitted job.

CPULOCK has also been added as a job template extension. The job template extension overrides the job script CPULOCK extension and the job submission CPULOCK option.

### *Improved Performance and Scalability*

Moab 8.0.0 includes new multi-threaded scheduling routines and is now compiled with optimizations enabled (-O). These improvements should increase scalability, efficiency, and performance for the majority of sites. The size of the thread pool can be throttled using the [THREADPOOLSIZE](#) parameter.

### *Finer-grained Logging Timestamps*

The timestamps in the Moab logs now include milliseconds, which can be helpful with higher [LOGLEVEL](#) settings.

### *Deleting checkpoint file doesn't reset IDs*

Moab now persists counters for job and reservation IDs to an external file (`<moab home>/counters`), in addition to persisting the counters to checkpoints. The new external counter file is created during installation. On startup, Moab reads the ID counters from either or both sources and uses the maximum of the two values that it sees as the resumption point for new IDs. This means that you can safely delete a checkpoint file without causing your IDs to be reset.

If both the external counter file and the checkpoint file are missing, Moab refuses to start. This is a precaution to protect the integrity of a relational database or external processes that rely on the IDs. If you need to override this behavior, you can do so by manually creating a new external counter file that starts IDs at an arbitrary number.

### *CANCELFAILEDDEPENDENCYJOBS scheduler flag*

The new [CANCELFAILEDDEPENDENCYJOBS](#) scheduler flag automatically cancels dependency jobs that will never run because of an unmet requirement. For more information, see [Job Dependencies](#).

### *Reduction in command processing time*

If your system's scheduling cycle regularly takes longer than the [CLIENTTIMEOUT](#) value, you can configure Moab to fork a copy of itself that will respond to certain information-only client commands (`checkjob`, `showbf`, and `showstart`). This enables you to run intense diagnostic commands while Moab is in the middle of its scheduling process. For more information, see [Reduce command processing time](#).

### *New --workflowjobids option for msub*

A new `--workflowjobids` option for the `msub` command returns all the job IDs, including data-staging system job IDs, in a data-staging workflow at submission time (Moab creates an internal workflow for

jobs submitted with data-staging options). For more information, see [Configuring data staging with advanced options](#).

### *NAMI interface improvements*

The Native Accounting Manager Interface now implements all charge policies and chargeable properties implemented by the MAM Accounting Manager Interface.

### *New support diagnostic script*

The `support.diag.pl` script used for gathering information for Adaptive support representatives has been deprecated with the 8.0.0 release. Use the new `support-diag.py` script instead. For more information, see [Diagnostic Scripts](#).

## Moab Web Services

### *PAM authentication support*

Support for PAM (pluggable authentication module) has been added to MWS. PAM treats the user as if it is local to the machine doing the authenticating, and it uses whatever the user is authenticating with, whether it be LDAP or NIS. For information about configuring PAM with MWS, see [PAM \(pluggable authentication module\) configuration using mws-config.groovy](#).



There is a security risk when authenticating local users through your PAM configuration. This behavior is highly discouraged and is not supported by Adaptive Computing.

### *OAuth authentication support*

Support for OAuth has been added to MWS. OAuth allows trusted client applications to securely delegate authentication to MWS. Once MWS has authenticated a user by verifying the username and password in LDAP, PAM, or NIS, MWS returns an access token to the client. The client then presents this access token to MWS to access resources. For information about configuring PAM with MWS, see [OAuth configuration using mws-config.groovy](#).

### *Node power management plugin*

The MWS power management plugin acts as a power resource manager; it monitors the power states of the nodes in the Node Configuration File using the TORQUE `pbsnodes` command and a query script. The plugin also allows you to customize the power state of each node when it should be "turned off." For more information, see the documentation included with the plugin in Moab Web Services.

### *Fairshare policy added to MWS Policies object*

The Moab "fairshare" policy has been added to the Policies object in MWS. With this addition, you can get fairshare policy information and make fairshare policy modifications through the MWS API. For more information, see .

### *Modifiable attributes added to the MWS job resource*

You can now use MWS to modify an idle job's system priority, the nodes that the job requests, and the resources per task required for the job. For more information, see [Modify job attributes](#).

### *View and modify single credentials*

You can now query a single credential in MWS. Additionally, you can modify a credential's fields and use the change-mode parameter to modify the credential's list fields. For more information, see [Credentials](#).

### *View and modify scheduler priorities*

You can now query and modify the scheduler priorities through the Priority resource in MWS. For more information, see [Priority](#).

## TORQUE Resource Manager

### *CPU frequency control*

TORQUE can now set the CPU frequency on requested nodes for submitted jobs. The request is made with the new resource extension. A user can specify a desired clock frequency in megahertz, a Linux power governor policy name, or an ACPI performance state (P-state) number.

The `qalter -l` command has been updated to allow modification of the requested CPU frequency on an already submitted job.

The `pass_cpuclock` server parameter was added which allows administrators to track, but not grant, the CPU frequency request portion of a job submission.

### *qrerun all command*

When you execute the `qrerun all` command, you will be prompted for confirmation. TORQUE will then place all running jobs in a queued state without contacting the MOMs. You should only use this when the entire cluster is down and you cannot contact it.

### *Node power state control*

TORQUE can now set the power state of a node. Depending on the hardware and software capabilities of the node, TORQUE can set the power state to Running, Standby, Suspend, Hibernate, or Shutdown. A [new option](#) was added to the `pbsnodes` command to make this work. The syntax of the command is:

```
pbsnodes -m [running|standby|suspend|hibernate|shutdown] <space delimited list of nodes to alter>
```

In order to wake a node from a low-power state, Wake-on-LAN must be supported and configured on the node. For more information, see [Changing node power states](#).

## Differences

This section contains differences in previously existing features that require a change in configuration or routine.

- [Moab HPC Suite on page 6](#)
- [Moab Workload Manager on page 6](#)
- [Moab Web Services on page 6](#)

- [TORQUE Resource Manager on page 6](#)
- [Differences on page 5](#)

## Moab HPC Suite

### PostgreSQL 9.3 repository

The PostgreSQL 9.3 repository information has been moved from the AC.repo file in /etc/yum.repos.d/ to its own file. Please make sure to use the newer AC.repo file to avoid duplicate repository entries for PostgreSQL 9.3.

## Moab Workload Manager

## Moab Web Services

## TORQUE Resource Manager

# Installation and upgrade information

## Installing Moab HPC Suite – Basic Edition 8.0.0

Please see [Installing the RPMs](#) for Basic for manual or RPM-based installation instructions, respectively.

## Installing TORQUE 5.0.0

### Installing TORQUE 5.0.0 on RHEL 5

TORQUE 5.0.0 requires the download of boost version 1.36 or later header files in order to build on most RHEL 5 installations. TORQUE 5.0.0 needs the `unordered_map` from the boost library. This did not become part of the boost standard until boost version 1.36. Standard RHEL 5 installations only have boost version 1.33.1. TORQUE 5.0.0 has added a configure option named `--with-boost-path`. This option allows the user to be able to download a newer version of boost without the need to install boost.

You can download a new boost library from [www.boost.org](http://www.boost.org). Extract the tarball to a directory of your choosing. When you are configuring TORQUE you then add the `--with-boost-path=<boost_path>` to the configuration line. The `boost_path` is the location where you unzipped the boost library. Within the unzipped files is a directory named `boost`. This directory contains the `.hpp` files with the needed

boost classes. Once the `--with-boost-path` option has been added to the configuration line you can continue to install TORQUE as per the installation guide.

Note that this procedure is not required on RHEL 6 and later versions.

## Upgrading to Moab HPC Suite – Basic Edition 8.0.0

**i** Upgrading an RPM-based installation of the Moab HPC Suite does not work correctly in 8.0.0. See [Known issues on page 7](#) for more information and tips.

Please see [Upgrading Moab Workload Manager](#) for Basic for upgrade instructions.

The Moab database schema has been updated for 8.0.0:

Table name	New additions	Changes
<b>Nodes</b>	<ul style="list-style-type: none"> <li>CPUClock (VARCHAR (64))</li> </ul>	<ul style="list-style-type: none"> <li>Column name "Partition" changed to "PartitionName". The word "partition" became a reserved word as of MySQL 5.6</li> </ul>
<b>Requests</b>	<ul style="list-style-type: none"> <li>CPUClock (VARCHAR (64))</li> </ul>	n/a
<b>NodeStats</b>	<ul style="list-style-type: none"> <li>CPUClock integer</li> <li>CPUMaxClock integer</li> <li>CPUMinClock integer</li> </ul>	n/a
<b>Reservations</b>	n/a	<ul style="list-style-type: none"> <li>Column name "Partition" changed to "PartitionName". The word "partition" became a reserved word as of MySQL 5.6</li> </ul>

To migrate your database schema to the newest version, see [Migrating Your Database to Newer Versions of Moab](#).

## Known issues

The following are known issues in Basic. Following each issue description is an associated issue number in parentheses.

- Connecting Moab to MySQL via ODBC can cause Moab to slow down. You can work around this issue by creating the tables with the old storage engine (ENGINE = MyISAM). Also note that other factors will also affect DB performance, such as the type of DB connection (remote vs. local), the DB server's hardware specifications, Moab's configuration for how much to write to the DB (see REALTIMEDBOBJECTS), etc (MOAB-6316).
- Sites that used RPMs to install Moab HPC Suite will not be able to upgrade to 8.0.0 RPMs until 7.2.9 is released. The issue exists in all pre-8.0 RPMs and cannot be worked around without uninstalling the suite. Thank you for your patience as we work to resolve these upgrade issues. Until the issues are addressed, we suggest a fresh installation (either [manual](#) or [RPM](#)) on a separate machine (MOAB-7192).
- Verification and validation of the Moab 8.0.0 integration with Oracle 12c is not yet complete (AC-7407).
- When you submit jobs to Moab with a proxy user but no group specified via MWS, Moab uses the root group rather than the group of the proxy user (WS-2111).
- When you submit jobs with MWS, you must set the commandFile field to the absolute path of the job script on the MWS server. MWS must have read access to the file given in commandFile. Submitting remote job scripts (that is, including the script as part of the JSON payload) is not currently supported (WS-2112).
- Jobs may linger in an exiting state on MOMs with 12 or more cores when multiple jobs start exiting at the same time. Eventually, the jobs clear but it may take several seconds or even minutes to clear them all (TRQ-2408).

## Resolved issues

The following is a list of some key bugs fixed in Basic. Following each issue description is an associated issue number in parentheses.

- **The scan\_for\_terminated action in TORQUE affected the speed of main\_loop.** The scan\_for\_terminated action in TORQUE has been placed on its own thread so that the MOM will be able to service incoming requests in a timely manner (TRQ-2408).
- **When the queue contained more than 50,000 jobs, TORQUE slowed down substantially.** TORQUE no longer slows down with a large number of jobs in the queue (TRQ-2345).
- **When pbs\_server had a high load, it would get stuck polling.** pbs\_server no longer gets stuck polling under these conditions (TRQ-2620).
- **When a job with a dependent job was deleted with qdel, TORQUE did not clean up the dependent job.** TORQUE now removes the dependent job when you delete its parent (TRQ-2621).
- **When a lot of jobs were run at once, one job would get stuck in an exiting state.** This error no longer occurs (TRQ-2622).
- **The afterok dependency did not work as expected.** afterok now works correctly (TRQ-2626).

- **When running a large amount of jobs, the server would crash during job recycle.** The crash no longer occurs (TRQ-2628).
- **pbsdsh required FQDNs even when they were working in pbs\_server.** The short names in pbs\_server are now enough to run pbsdsh (TRQ-2632).
- **A deadlock would occur on job\_save failure.** This error no longer occurs (TRQ-2645).
- **Asynchronous job starts queued in TORQUE but not yet serviced caused Moab to reschedule jobs that would eventually run.** A new job sub-state in TORQUE prevents this from occurring (TRQ-2715).

## Key documentation changes

This section lists some key documentation changes made for Moab HPC Suite – Basic Edition 8.0.0. Following each change description is a link to the affected page and an associated issue number if applicable.

Features new to this release have been documented but are not listed on this page. For information about those new features and links to their respective documentation, see [New features on page 1](#).

- [Moab Suite on page 9](#)
- [TORQUE Resource Manager on page 9](#)

### Moab Suite

#### [Additions](#)

#### [Removals](#)

- The "Scheduling Jobs When VMs Exist" topic has been removed from Prioritizing Jobs and Allocating Resources. Obsolete feature (DOC-1905)
- The VMUSAGE as been removed from the "Cloud-specific job template attributes" topic. Obsolete feature (DOC-1905).

#### [Improvements](#)

### TORQUE Resource Manager

#### [Additions](#)

#### [Improvements](#)

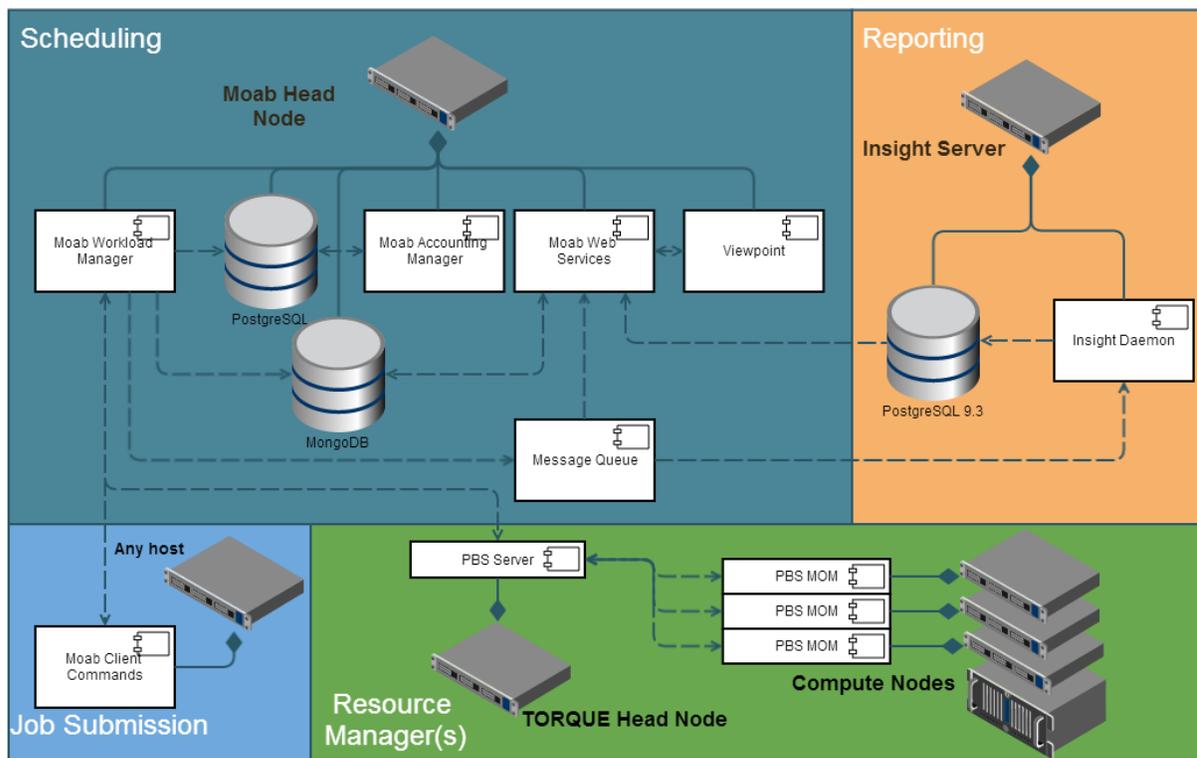


# Installation and Configuration

## Requirements

There are many different ways to install and configure Moab HPC Suite. Each environment has its own set of requirements and preferences. The following installation instructions are intended to help an administrator understand how each of the Moab HPC Suite components interact and how to install and configure each one. Two approaches have been documented: the “Manual installation” and the “RPM installation”. Only one approach is required for installation; do not try to follow both sets of instructions on a single system.

The diagram below provides a general topology of the Moab HPC Suite.



Please note the following:

- Moab Accounting Manager is available only with the Moab HPC Enterprise Suite.
- Smaller environments may elect to consolidate the TORQUE Head Node with the Moab Head Node, including PBS Server in the list of components installed on the Moab Head Node.
- The Requirements section gives further clarification regarding what each component requires.
- Although Moab Workload Manager and Moab Accounting Manager may share the same database instance, it is not a requirement. Two database instances may be used, one for each component. See the Requirements section for more information about what databases are supported.
- Larger systems will require more dedicated resources for each component, in which case it may be necessary to move individual components from the Moab Head Node (i.e. databases, Moab Accounting Manager, and/or Viewpoint) to their own respective servers.
- The Message Queue component is fulfilled by [ZeroMQ™](#). The libraries are provided with the components that use the message queue and are enabled via configuration; no special installation is necessary.

## Where to Start

1. Begin by reading the Requirements section below. Whether installing manually or with RPMs, it is important to be familiar with the hardware and software requirements.
2. Decide whether you will perform a manual installation or an RPM installation.

The manual installation provides advantages to administrators who want to pick and choose what components to install and administrators who want non-standard configure options. The RPM installation provides advantages to administrators who want a fairly standard installation.

 Code samples have been provided for convenience. Some code samples provide sample passwords (i.e. “changeme!”). We strongly recommend that you do not use these passwords during installation, as using the documented passwords could introduce unnecessary security vulnerabilities into your system.

Then follow the appropriate installation instructions.

3. The “Additional Configuration” section in both the manual and the RPM installation instructions provide additional information and instructions for optional, but recommended configurations (i.e. Configuring SSL in Tomcat, etc.).
4. Refer to [Troubleshooting on page 78](#) for assistance in addressing common problems during installation and configuration.
5. Refer to [Component documentation on page 86](#) for links to additional administrator and reference guides.

## Requirements

### Moab HPC Suite

#### Hardware Requirements

The following are the minimum hardware requirements for an average environment. Larger environments should consider allocating more resources and/or spreading components across multiple servers. Please consult the table below for recommendations:

Type	# of Compute Nodes	Jobs/week	Minimum Requirements (combine TORQUE & Moab head nodes on one server)	Recommended Requirements (targeting minimum number of servers)
Proof of Concept / Small Demo	50	<1k	<b>Moab+TORQUE Head Node:</b> <ul style="list-style-type: none"> <li>• 4 Intel/AMD x86-64 cores</li> <li>• At least 8 GB RAM</li> <li>• At least 100 GB dedicated disk space</li> </ul>	Same as minimum
Medium	500	<100k	<b>Moab+TORQUE Head Node:</b> <ul style="list-style-type: none"> <li>• 8 Intel/AMD x86-64 cores</li> <li>• At least 16 GB RAM</li> <li>• At least 512 GB dedicated disk space</li> </ul>	<b>Moab+TORQUE Head Node:</b> <ul style="list-style-type: none"> <li>• 16 Intel/AMD x86-64 cores</li> <li>• At least 32 GB RAM</li> <li>• At least 1 TB dedicated disk space</li> </ul>

Type	# of Compute Nodes	Jobs/week	Minimum Requirements (combine TORQUE & Moab head nodes on one server)	Recommended Requirements (targeting minimum number of servers)
Medium with High Throughput or Larger	>500	>100k	<p><b>Moab Head Node:</b></p> <ul style="list-style-type: none"> <li>• 8 Intel/AMD x86-64 cores</li> <li>• At least 16 GB RAM</li> <li>• At least 512 GB dedicated disk space</li> </ul> <p><b>TORQUE Head Node:</b></p> <ul style="list-style-type: none"> <li>• 8 Intel/AMD x86-64 cores</li> <li>• At least 16 GB RAM</li> <li>• At least 512 GB dedicated disk space</li> </ul>	We recommend separating components onto separate servers where possible (some components should not be separated; see Requirements below). Specific requirements around the intended configuration and use of Moab HPC Suite will help determine suite topology and resource allocation.

Please note the following:

- All requirements above (minimum and recommended) target a minimum number of management servers. Administrators are encouraged to separate the TORQUE and Moab head nodes where possible for better results, especially when High Throughput is enabled.
- Although many factors may have an impact on performance (network bandwidth, intended use and configuration, etc.), we consider High Throughput as something that makes a significant enough difference between minimum and recommended hardware requirements to merit mention in the table above.
- Moab and TORQUE are both multi-threaded and perform better with more processors.
- Regarding disk space, consideration should be given to requirements related to log files, log depth, number of jobs/nodes/reservations (more objects impact database journal size), average number of events generated (more events take more space), etc.

### *Software Requirements*

The installation documentation provides more details regarding how to install and configure the following software requirements. The information provided below is for your information only. No action is necessary.

Software requirements are listed per-component rather than suite-wide to make it easier for administrators who wish to install components on separate servers.

## **TORQUE**

### *Supported Operating Systems*

- CentOS 6.5 or later
- Red Hat 6.5 or later
- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later

**i** CentOS 5.9, Red Hat 5.9 and Scientific Linux 5.9 are supported, largely to continue support for clusters where the compute nodes operating systems cannot be upgraded. We recommend that the TORQUE head node run on the supported operating systems listed above.

### *Software Requirements*

- libxml2-devel package (package name may vary)
- openssl-devel package (package name may vary)
- Tcl/Tk version 8 or later if you plan to build the GUI portion of TORQUE or use a Tcl based scheduler
- If you use [cpuset](#), libhwloc 1.1 or later is required (for TORQUE 4.0.0 and later)

If you build TORQUE from source (i.e. clone from github), the following additional software is required:

- gcc
- gcc-c++
- A posix compatible version of make
- libtool 1.5.22
- boost-devel 1.36.0

## **Moab Workload Manager**

### *Supported Operating Systems*

- CentOS 6.5 or later
- Red Hat 6.5 or later

- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later

#### *Software Requirements*

- [libcurl](#)
- Perl 5.8.8 or later
- perl-cpan (package name may vary)
- libxml2-devel (package name may vary)
- *(Optional)* MySQL, PostgreSQL, or Oracle with ODBC driver (see [Database Configuration on page 793](#) for details)

#### *Supported Resource Managers*

- TORQUE 5.0.0
- SLURM

### **Moab Accounting Manager**

MAM is commonly installed on the same host as Moab Workload Manager; however, in some cases you might obtain better performance by installing them on separate hosts.

#### *Supported Operating Systems*

- CentOS 6.5 or later
- Red Hat 6.5 or later
- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later

#### *Software Requirements*

- gcc
- perl-suidperl
- httpd
- mod\_ssl
- rrdtool
- Perl modules (see Installing for more details)

#### *Depends On (not necessarily on the same server)*

MAM uses an RDBMS as a back end.

- Moab® Workload Manager 8.0.0
- PostgreSQL 7.2 or later

## Moab Web Services

MWS should be installed on the same host as Moab® Workload Manager.

### Supported Operating Systems

- CentOS 6.5 or later
- Red Hat 6.5 or later
- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later

### Software Requirements

- Moab® Workload Manager 8.0.0
- Apache Tomcat™ 6
- Oracle® Java® 7 Runtime Environment

**i** Oracle Java 7 Runtime Environment is the recommended Java environment, but Oracle Java 6 is also supported. All other versions of Java, including OpenJDK/IcedTea, GNU Compiler for Java, and so on cannot run Moab Web Services.

- MongoDB® 2.4.x

*Depends On (not necessarily on the same server)*

- OpenLDAP or PAM (see [Configuring Moab Web Services on page 1395](#))

# Manual installation

## Installation

### Preparing for installation

The installation process of the Moab HPC Suite includes installing the separate components in the suite. This guide contains detailed instructions for installing each component.

**i** Many individual components have dependencies on other components (see [Requirements on page 11](#)). However, if you do not require a certain component (Moab Web Services, for example), you do not have to install it.

The install instructions for each component include information about system requirements and dependencies. Some include prerequisite instructions that you will need to complete before you begin the install. Please read this information carefully, and make sure you have installed all the dependencies and packages that are necessary in order to avoid errors during the Moab HPC Suite install process.

**i** Because many system-level files and directories are accessed during the installation, the instructions in this guide should be executed with root privileges.

You will see that the instructions execute commands as the root user. Please note that the same commands will work for a non-root user with the `sudo` command.

To install the Moab HPC Suite, install the packages in the following order:

1. Install the TORQUE and Moab Workload Manager dependencies (see the [dependency installation instructions](#) below).
2. Install TORQUE (see [Installing TORQUE on page 2544](#)).
3. Install Moab Workload Manager (see [Installing Moab Workload Manager on page 27](#)).
4. Install Moab Web Services (see [Installing Moab Web Services on page 31](#)).

**i** Once you have finished installing the Moab HPC Suite and Moab is running, you can perform an RPM-based installation of the optional Insight and Viewpoint components.

## Install TORQUE and Moab Workload Manager dependencies

You must install the following dependencies in order to use TORQUE and Moab Workload Manager:

- libxml2-devel package (package name may vary)
- openssl-devel package (package name may vary)
- boost-devel package (package name may vary)
- ANSI C compiler. The native C compiler is recommended if it is ANSI; otherwise use `gcc`.

Use the following commands to install the required dependencies and packages.

- **RHEL, CentOS, and Scientific Linux:**

```
[root]# yum install make perl-CPAN libxml2-devel openssl-devel boost-devel gcc gcc-c++
```

- **SLES:**

Before installing the dependencies, do the following:

1. Verify that you have a licensed installation of SLES 11 SP3.
2. Download the [SuSE Linux Enterprise 11 Software Development Kit e-Media Kit](#) and add the ISO to the repository.

```
[root]# zypper install make libxml2-devel libopenssl-devel boost-devel gcc gcc-c++ git-core automake
```

## Install Java

You must install the 64-bit RPM version of Oracle® Java® 7 Runtime Environment if you are installing Moab Web Services.

**i** Oracle Java 7 Runtime Environment is the recommended Java environment, but Oracle Java 6 is also supported. All other versions of Java, including OpenJDK/IcedTea, GNU Compiler for Java, and so on cannot run Moab Web Services.

Do the following:

- Download the *Linux x64 RPM* version of Oracle Java SE 7 JRE. (Go to the [Oracle Java 7 download page](#), copy the URL to the *Linux x64 RPM* version, then run the following command.)

```
[root]# wget <URL> -O jre-7-linux-x64.rpm
```

To verify that the download was successful, run the following on the RPM before installation:

```
[root]# rpm -qip jre-7-linux-x64.rpm
```

- Run the following to install Java 7:

```
[root]# rpm -Uh jre-7-linux-x64.rpm
```

## Install Tomcat

You must install Tomcat if you are installing Moab Web Services.

- **RHEL, CentOS, and Scientific Linux:**

```
[root]# yum install tomcat6
```

- **SLES:**

```
[root]# zypper ar --refresh -r
http://download.opensuse.org/evergreen/11.4/openSUSE:Evergreen:11.4.repo
[root]# zypper in tomcat6
[root]# zypper mr -d openSUSE_Evergreen_11.4
```

## Opening ports

A few ports need to be available through your firewall so components of the suite can communicate with each other. Some features of some components might need additional ports configured. The individual component documentation indicates when additional ports are needed.

The ports required for basic suite functionality are:

- 7112: Default, configurable port needed for Moab Accounting Manager client-server communication
- 443: Needed for Moab Accounting Manager web GUI (https)
- 8080: Needed for Moab Web Services web portal (http)

## To open ports in your firewall

- **Use iptables for Red Hat-based distributions:**

```
[root]# iptables-save > /tmp/iptables.mod
[root]# vi /tmp/iptables.mod

# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 7112 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
-A INPUT -p tcp --dport 8080 -j ACCEPT

[root]# iptables-restore < /tmp/iptables.mod
[root]# service iptables save
```

- **Use SuSEfirewall2 for SuSE-based distributions:**

```
[root]# vi /etc/sysconfig/SuSEfirewall2

FW_SERVICES_EXT_TCP="443 7112 8080"

[root]# service SuSEfirewall2_setup restart
```

## Install MongoDB

You must install MongoDB if you are installing Moab Web Services.

### To install and enable MongoDB

1. Install MongoDB.

- **RHEL and CentOS, and Scientific Linux:**

Create a file called `/etc/yum.repos.d/10gen.repo` and add the following lines.

```
[10gen]
name=MongoDB Repository
baseurl=http://downloads-distro.mongodb.org/repo/redhat/os/x86_64
gpgcheck=0
enabled=1
```

Install `mongo-10gen-server`.

```
[root]# yum install mongo-10gen-server --exclude mongodb-org,mongodb-org-server
```

- **SLES:**

```
[root]# zypper ar
http://download.opensuse.org/repositories/server:/database/SLE_11_SP3
OpenSuseDatabase
[root]# zypper install mongodb
```

## 2. Start MongoDB.

- **RHEL and CentOS, and Scientific Linux:**

```
[root]# chkconfig mongod on
[root]# service mongod start
```

- **SLES:**

```
[root]# chkconfig mongod on
[root]# service mongod start
```

**i** There may be a short delay (approximately 3 minutes) for Mongo to start the first time.

**i** If you see errors while running the `chkconfig` command, make sure that `/sbin` is in your `PATH` environment variable, then run `chkconfig` again.

```
export PATH=/sbin:$PATH
```

## 3. Prepare the MongoDB database by doing the following:

## a. Add the required MongoDB users.

**i** The passwords used below (`secret1`, `secret2`, and `secret3`) are examples. Choose your own passwords for these users.

```
[root]# mongo
> use admin;
> db.addUser("admin_user", "secret1");
> db.auth ("admin_user", "secret1");

> use moab;
> db.addUser("moab_user", "secret2");
> db.addUser("mws_user", "secret3", true);

> use mws;
> db.addUser("mws_user", "secret3");
> exit
```

**i** Because the `admin_user` has read and write rights to the `admin` database, it also has read and write rights to all other databases. See [Control Access to MongoDB Instances with Authentication](#) for more information.

## b. Enable authentication in MongoDB.

- **RHEL and CentOS, and Scientific Linux:**

```
[root]# vi /etc/mongod.conf

auth = true

[root]# service mongod restart
```

- **SLES:**

```
[root]# vi /etc/mongodb.conf
auth = true

[root]# service mongodb restart
```

 On SLES machines, auth = true is enabled by default.

## Install PostgreSQL

### To install PostgreSQL

1. Install and initialize PostgreSQL.

#### CentOS, RHEL, and Scientific Linux

```
[root]# yum install postgresql-server
[root]# service postgresql initdb
```

#### SLES

```
[root]# zypper install postgresql-server
[root]# service postgresql start
```

2. Configure trusted connections.

 Edit or add a "host" line in the pg\_hba.conf file for the interface from which the server(s) (for example, Moab Workload Manager and/or Moab Accounting Manager) will be connecting to the database and ensure that it specifies a secure password-based authentication method (for example, md5).

```
[root]# vi /var/lib/pgsql/data/pg_hba.conf

# IPv4 local connections:
host    all             all             127.0.0.1/32      md5
# IPv6 local connections:
host    all             all             :::1/128         md5
```

3. Configure PostgreSQL to accept connections from your host.

```
[root]# vi /var/lib/pgsql/data/postgresql.conf

# Uncomment the listen addresses line in the configuration:

listen_addresses = 'localhost'          # what IP address(es) to listen on;
```

4. Start or restart the database.

```
[root]# chkconfig postgresql on
[root]# service postgresql restart
```

## Related topics

- [Welcome on page xxiii](#)

# Installing TORQUE

These instructions describe how to install and start TORQUE.

## Requirements

### *Supported Operating Systems*

- CentOS 6.5 or later
- Red Hat 6.5 or later
- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later



CentOS 5.9, Red Hat 5.9 and Scientific Linux 5.9 are supported, largely to continue support for clusters where the compute nodes operating systems cannot be upgraded. We recommend that the TORQUE head node run on the supported operating systems listed above.

### *Software Requirements*

- libxml2-devel package (package name may vary)
- openssl-devel package (package name may vary)
- Tcl/Tk version 8 or later if you plan to build the GUI portion of TORQUE or use a Tcl based scheduler
- If you use [cpuset](#), libhwloc 1.1 or later is required (for TORQUE 4.0.0 and later)

If you build TORQUE from source (i.e. clone from github), the following additional software is required:

- gcc
- gcc-c++
- A posix compatible version of make
- libtool 1.5.22
- boost-devel 1.36.0

## Prerequisites

- TORQUE requires certain ports to be open for essential communication:
  - For client communication to `pbs_server`, all privileged ports must be open (ports under 1024).
  - For `pbs_server` communication to `pbs_mom`, the default port is 15003.
  - For `pbs_mom` to `pbs_server`, the default port is 15001.

For more information on how to configure the ports that TORQUE uses for communication, see [Configuring ports on page 2555](#).

**i Important:** If you intend to use TORQUE 5.0.0 with Moab, you must run Moab version 8.0.0 or later. TORQUE 5.0.0 will not work with versions earlier than Moab 8.0.0.

- Make sure your host (with the correct IP address) is in your `/etc/hosts` file.
- The `libxml2-devel`, `openssl-devel`, and `boost-devel` packages must be installed (These packages should already be installed from following the steps in the [Preparing for installation on page 17](#)).

### RHEL 6.5 and CentOS 6.5, and Scientific Linux 6.5:

```
[root]# yum install openssl-devel libtool-devel libxml2-devel boost-devel gcc gcc-c++
```

### SLES

```
[root]# zypper install openssl-devel libtool-devel libxml2-devel boost-devel gcc gcc-c++
```

### RHEL 5 and CentOS 5, and Scientific Linux 5:

```
[root]# yum install openssl-devel libtool-devel libxml2-devel gcc gcc-c++
```

**i Important:** TORQUE requires Boost version 1.36.0 or greater. The `boost-devel` package provided with RHEL 5, CentOS 5, and Scientific Linux 5 is older than this requirement. A new option, `--with-boost-path` has been added to configure (see [Customizing the install on page 2558](#) in the *TORQUE Administrator Guide* for more information). This allows you to point TORQUE to a specific version of boost during make. One way to compile TORQUE without installing Boost is to simply download the Boost version you plan to use from: <http://www.boost.org/users/history/>. Next, untar Boost—you do not need to build it or install it. When you run TORQUE configure, use the `--with-boost-path` option pointed to the extracted Boost directory.

## To install TORQUE

1. Switch the user to root.

```
[user]$ su -
```

2. Download the latest 5.0.0 build from the [Adaptive Computing](#) website. It can also be downloaded via command line.
  - a. Clone the source from github. If you clone the source from github, the `libtool` package must be installed.

**i** If you are using CentOS 5, use these instructions for installing `libtool`:

```
[root]# cd /tmp
[root]# wget http://ftpmirror.gnu.org/libtool/libtool-2.4.2.tar.gz
[root]# tar -xzvf libtool-2.4.2.tar.gz
[root]# cd libtool-2.4.2
[root]# ./configure --prefix=/usr
[root]# make
[root]# make install
[root]# cd /tmp
[root]# git clone https://github.com/adaptivecomputing/torque.git -b
5.0.0 5.0.0
[root]# cd 5.0.0
[root]# ./autogen.sh
```

```
# RHEL 6 and Scientific Linux 6:
[root]# yum install git libtool
```

```
# SLES:
[root]# zypper install libtool
```

```
[root]# git clone https://github.com/adaptivecomputing/torque.git -b 5.0.0 5.0.0
[root]# cd 5.0.0
[root]# ./autogen.sh
```

- b. Get the tarball source distribution.

```
[root]# wget http://www.adaptivecomputing.com/download/torque/torque-
5.0.0.tar.gz -O torque-5.0.0.tar.gz

[root]# tar -xzvf torque-5.0.0.tar.gz
[root]# cd torque-5.0.0/
```

3. Run each of the following commands in order.

```
[root]# ./configure
[root]# make
[root]# make install
```

For information on what options are available to customize the `./configure` command, see [Customizing the install on page 2558](#).

4. Configure the `trqauthd` daemon to start automatically at system boot.

```

* If RHEL distribution, do the following *
[root]# cp contrib/init.d/trqauthd /etc/init.d/
[root]# chkconfig --add trqauthd
[root]# echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf
[root]# ldconfig
[root]# service trqauthd start

* If SLES distribution, do the following *
[root]# cp contrib/init.d/suse.trqauthd /etc/init.d/trqauthd
[root]# chkconfig --add trqauthd
[root]# echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf
[root]# ldconfig
[root]# service trqauthd start

```

5. The `make packages` command can be used to create self-extracting packages that can be copied and executed on your nodes. For information on creating packages and deploying them, see [Compute nodes on page 2549](#).

You will also want to scp the `init.d` scripts to the compute nodes and install them there.

6. Verify that the `/var/spool/torque/server_name` file exists and contains the correct name of the server.

```
[root]# echo <pbs_server's_hostname> > /var/spool/torque/server_name
```

7. By default, TORQUE installs all binary files to `/usr/local/bin` and `/usr/local/sbin`. Make sure the path environment variable includes these directories for both the installation user and the root user.

```
[root]# export PATH=/usr/local/bin:/usr/local/sbin:$PATH
```

8. Initialize `serverdb` by executing the `torque.setup` script.

```
[root]# ./torque.setup root
```

9. Add nodes to the `/var/spool/torque/server_priv/nodes` file. For information on syntax and options for specifying compute nodes, see [Specifying compute nodes on page 2553](#).
10. Configure the MOMs if necessary (see [Configuring TORQUE on compute nodes on page 2554](#) in the TORQUE Administrator Guide).
11. Configure `pbs_server` and `pbs_mom` to start automatically at system boot, and then start their daemons.

```

* If RHEL distribution, do the following *
[root]# cp contrib/init.d/pbs_server contrib/init.d/pbs_mom /etc/init.d
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom
[root]# service pbs_server restart
[root]# service pbs_mom start

* If SLES distribution, do the following *
[root]# cp contrib/init.d/suse.pbs_server /etc/init.d/pbs_server
[root]# cp contrib/init.d/suse.pbs_mom /etc/init.d/pbs_mom
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom
[root]# service pbs_server restart
[root]# service pbs_mom start

```

### Related topics

- [Preparing for installation on page 17](#)
- [Installing Moab Workload Manager on page 27](#)
- [Component documentation on page 86](#)

## Installing Moab Workload Manager

These instructions describe how to install and start Moab Workload Manager (Moab).

### Dependencies and packages installation

Use the following commands to install the required Moab Workload Manager dependencies and packages (listed in the [Installing Moab Workload Manager](#) section above).

#### RHEL, CentOS, and Scientific Linux:

```

[root]# yum update
[root]# yum install make libcurl perl-CPAN libxml2-devel

```

#### SLES:

```

[root]# zypper update
[root]# zypper install make curl libxml2-devel

```

### Optional: To build a custom RPM

1. Install rpm-build.

```

[root]# yum install rpm-build

```

2. Download the latest Moab build (moab-*<version>*-*<OS>*.tar.gz) from the [Adaptive Computing website](#).

**i** The variable marked *<version>* is the desired version of the suite; for example, 8.0-2014061017-8f96ac8d would be Moab 8.0 revision 2014061017 at changeset 8f96ac8d. The variable marked *<OS>* indicates which OS the build was designed for.

3. Untar the downloaded package.

4. Change directories into the untarred directory.
5. Edit the `./moab.spec` file for RPM customization.
6. Run `./rpm-build`.
7. Locate the custom RPM in `rpm/RPMS/x86_64`.

## To install Moab Workload Manager

1. Download the latest Moab build (`moab-<version>-<OS>.tar.gz`) from the [Adaptive Computing website](#).

**i** The variable marked `<version>` is the desired version of the suite; for example, `8.0-2014061017-8f96ac8d` would be Moab 8.0 revision 2014061017 at changeset 8f96ac8d. The variable marked `<OS>` indicates which OS the build was designed for.

2. As the root user, run each of the following commands in order.

```
[root]# tar xzvf moab-<version>-<OS>.tar.gz
[root]# cd moab-<version>-<OS>
```

3. Configure Moab. For a complete list of `./configure` options, use `./configure --help` or refer to [Moab Workload Manager configuration options on page 42](#) for a list of commonly used options.

It is strongly recommended that you configure Moab with the `--with-init` and `--with-profile` options. The `--with-profile` option makes it easier to execute Moab commands (see step 8). The `--with-init` option allows Moab to automatically start at OS startup (see step 11).

```
[root]# ./configure <options>
```

4. (Only if you are using green computing, or if you are using a resource manager other than TORQUE) Run the `make perldeps` command to install the necessary perl modules using CPAN. When first running CPAN, you will be asked for configuration information. It is recommended that you choose an automatic configuration. You will be prompted to provide input during module installation; running the `make perldeps` command with a script is not recommended.

```
[root]# make perldeps
```

5. Install Moab.

```
[root]# make install
```

6. (ONLY if installing on non-RHEL distributions)

Copy the appropriate `init.d` file, set the permissions on it, and configure Moab to start automatically at system boot.

```
* If SLES distribution, do the following *
[root]# cp OS/SLES/etc/init.d/moab /etc/init.d/moab

[root]# chmod 755 /etc/init.d/moab
[root]# chkconfig --add moab
```

7. Modify the Moab configuration file.

```
[root]# vi /opt/moab/etc/moab.cfg
```

Do the following:

- a. Verify that **SUBMITCMD** is set up for your TORQUE resource manager and that it points to a valid qsub executable. For example:

```
RMCFG[torque] SUBMITCMD=/usr/local/bin/qsub
```

If you use a SLURM resource manager, see [Moab-SLURM Integration Guide](#) for configuration information. If you use a NATIVE resource manager, see [Managing Resources Directly with the Native Interface](#) for configuration information.

- b. *ONLY* if you are using Moab Web Services, add *tomcat* to the list of administrator **USERS**. For example:

```
ADMINCFG[1] USERS=root,tomcat
```

8. If you configured with the `./configure --with-profile` option, source the following file to add the Moab executable directories to your current shell `$PATH` environment.

```
[root]# . /etc/profile.d/moab.sh
```

9. Copy your license file into the same directory as `moab.cfg` (`/opt/moab/etc/` by default). For example:

```
[root]# cp moab.lic $MOABHOMEDIR/etc/moab.lic
```

To verify the current status of your license, use `moab --about`.

Moab checks the status of the license every day just after midnight. At 60 and 45 days before, and daily from 30 days before license expiration to and including the license expiration date, Moab sends an e-mail to all level 1 administrators informing them of the pending Moab license expiration. A log record is also made of the upcoming expiration event. For the notifications to occur correctly, you must enable administrator email notification (see "[Notifying Administrators of Failures on page 684](#)" in the Moab Workload Manager Administrator Guide) and `moab.cfg` must contain email addresses for level 1 administrators. For example:

```
ADMINCFG[1] USERS=u1,u2,u3[,...]

USERCFG[u1] EMAILADDRESS=u1@company.com
USERCFG[u2] EMAILADDRESS=u2@company.com
USERCFG[u3] EMAILADDRESS=u3@company.com

MAILPROGRAM DEFAULT
```

**i** Moab will not run without a license. For information about obtaining a trial license, please contact [Adaptive Computing](#).

10. Start Moab.

```
[root]# chkconfig moab on
[root]# service moab start
```

11. Submit a sleep job as a non-root user and verify the job is running.

```
[root]# su - user
[user]$ echo sleep 150 | msub
[user]$ showq
[user]$ exit
```

12. Connecting Moab to MongoDB

If you will be installing Moab Web Services, connect Moab to MongoDB using the following instructions:

 The `USEDATABASE` parameter is unrelated to the MongoDB configuration.

- a. Set the **MONGOSERVER** parameter in `/opt/moab/etc/moab.cfg` to the MongoDB server hostname. Use localhost as the hostname if Moab and MongoDB are hosted on the same server.

```
MONGOSERVER <host>[:<port>]
```

If your **MONGOSERVER** host is set to anything other than localhost, edit the `/etc/mongod.conf` file on the MongoDB server host and either comment out any `bind_ip` parameter or set it to the correct IP address:

```
# Listen to local interface only. Comment out to listen on all interfaces.
#bind_ip=127.0.0.1
```

- b. In the `/opt/moab/etc/moab-private.cfg` file, set the **MONGOUSER** and **MONGOPASSWORD** parameters to the MongoDB `moab_user` credentials you set (for details, see [Install MongoDB on page 20](#)).

```
MONGOUSER      moab_user
MONGOPASSWORD  secret2
```

- c. Verify that Moab is able to connect to MongoDB.

```
[root]# service moab restart
[root]# mdiag -S
...
Mongo connection (localhost) is up (credentials are set)
...
```

13. Securing communication using secret keys

- a. (required) Moab and MWS use Message Authentication Codes (MAC) to ensure messages have not been altered or corrupted in transit. Generate a key and store the result in `/opt/moab/etc/.moab.key`:

```
[root]# service moab stop
[root]# dd if=/dev/urandom count=18 bs=1 2>/dev/null | base64 >
/opt/moab/etc/.moab.key
[root]# chown root:root /opt/moab/etc/.moab.key
[root]# chmod 400 /opt/moab/etc/.moab.key
[root]# service moab start
```

The key you specify in the `.moab.key` file is the same key you must also specify in the `moab.secretKey` property when installing and configuring MWS (see [Installing Moab Web Services on page 31](#)).

- b. (optional) Moab supports message queue security using AES. This feature requires a Base64-encoded 16-byte (128-bit) shared secret. Generate a key and append the result to `/opt/moab/etc/moab-private.cfg`:

```
[root]# service moab stop
[root]# echo "MESSAGEQUEUESECRETKEY $(dd if=/dev/urandom count=16 bs=1
2>/dev/null | base64)" >> /opt/moab/etc/moab-private.cfg
[root]# service moab start
```

The key you specify in the `moab-private.cfg` file is the same key you must also specify in the `moab.messageQueue.secretKey` property when installing and configuring MWS (see [Installing Moab Web Services](#)).

**i** If MWS is configured to encrypt the message queue and Moab is not (or vice versa), then MWS will ignore the messages from Moab. Furthermore, all attempts to access the MWS service resource will fail.

- c. (optional) Verify that encryption is on for the ZeroMQ connection.

```
[root]# mdiag -S|grep 'ZeroMQ MWS'
ZeroMQ MWS connection is bound on port 5570 (encryption is on)
```

#### Related topics

- [Preparing for installation on page 17](#)
- [Installing TORQUE on page 2544](#)
- [Component documentation on page 86](#)

## Installing Moab Web Services

These instructions describe how to install Moab Web Services (MWS).

### To install Moab Web Services

**i** You must deploy Moab Web Services on the same server as Moab Workload Manager.

1. Verify Moab is installed and configured as desired (for details, see [Installing Moab Workload Manager](#)).
2. Start Moab.

```
[root]# service moab start
```

3. Create the MWS home directory and subdirectories (for more information, see the "[Configuration on page 1772](#)" section of the Moab Web Services Reference Guide).

**i** The default location for the MWS home directory is `/opt/mws`. These instructions assume the default location.

Here is a sample script for this setup:

```
[root]# mkdir -p \
/opt/mws/etc/mws.d \
/opt/mws/hooks \
/opt/mws/log \
/opt/mws/plugins \
/opt/mws/spool/hooks \
/opt/mws/utils
[root]# chown -R tomcat:tomcat /opt/mws # Depending on your OS, the Tomcat username
might be tomcat6.
[root]# chmod -R 555 /opt/mws
[root]# chmod u+w \
/opt/mws/log \
/opt/mws/plugins \
/opt/mws/spool \
/opt/mws/spool/hooks \
/opt/mws/utils
```

4. Download the latest MWS build (`mws-<version>.tar.gz`) from the [Adaptive Computing](#) website.
5. Extract the contents of the MWS download tarball into a temporary directory. For example:

```
[root]# mkdir /tmp/mws-install
[root]# cd /tmp/mws-install
[root]# tar xvzf $HOME/Downloads/mws-8.0.0.tar.gz
```

6. Copy the extracted utility files to the utility directory created above and give the tomcat user ownership of the directory.

```
[root]# cd /tmp/mws-install/mws-8.0.0/utils
[root]# cp * /opt/mws/utils
[root]# chown tomcat:tomcat /opt/mws/utils/*
```

7. Set up the MWS configuration files. In the extracted directory are several configuration files.

- a. Copy `mws-config.groovy` to `/opt/mws/etc`.

```
[root]# cd /tmp/mws-install/mws-8.0
[root]# cp mws-config.groovy /opt/mws/etc
```

- b. Copy the appropriate suite-specific file to `/opt/mws/etc/mws.d`. Pick from the files matching filename `mws-config-*.groovy` in `/tmp/mws-install/mws-8.0`.

```
[root]# cp mws-config-<your suite choice>.groovy /opt/mws/etc/mws.d
```

- c. Give the Tomcat user read access to `/opt/mws/etc/mws-config.groovy` and `/opt/mws/etc/mws.d/mws-config-*.groovy`.

```
[root]# chown tomcat:tomcat /opt/mws/etc/mws-config.groovy
[root]# chown tomcat:tomcat /opt/mws/etc/mws.d/mws-config-<your suite
choice>.groovy
[root]# chmod u+r /opt/mws/etc/mws-config.groovy
[root]# chmod u+r /opt/mws/etc/mws.d/mws-config-<your suite choice>.groovy
```

d. In the `/opt/mws/etc/mws-config.groovy` file, change these settings:

- **moab.secretKey:** Must match the Moab secret key you generated earlier (contained in `/opt/moab/etc/.moab.key`).
- **auth.defaultUser.username:** Any value you like, or leave as is.
- **auth.defaultUser.password:** Any value you like, but choose a strong password.
- **moab.messageQueue.secretKey:** Add this property to configure the message queue security key in MWS.

 The key you specify must be encoded in [Base64](#), and must match *exactly* the key specified in the **MESSAGEQUEUESECRETKEY** when installing Moab Workload Manager (see [Installing Moab Workload Manager](#) on page 27).

 Important: If MWS is configured to encrypt the message queue and Moab is not (or vice versa) then the messages from Moab will be ignored. Furthermore, all attempts to access the MWS service resource will fail.

```
[root]# vi /opt/mws/etc/mws-config.groovy

// Replace <ENTER-KEY-HERE> with the contents of /opt/moab/etc/.moab.key.
moab.secretKey = "<ENTER-KEY-HERE>"
moab.server = "localhost"
moab.port = 42559

// Replace <ENTER-KEY-HERE> with the value of MESSAGEQUEUESECRETKEY in
/opt/moab/etc/moab-private.cfg.

moab.messageQueue.secretKey = "<ENTER-KEY-HERE>"

// Change these to be whatever you like.

auth.defaultUser.username = "moab-admin"
auth.defaultUser.password = "changeme!"
```

 If you do not change **auth.defaultUser.password**, your MWS will not be secure (because anyone reading these instructions would be able to log into your MWS). Here are some [tips](#) for choosing a good password.

e. Do *one* of the following:

**i** You can configure only one authentication method in `mws-config.groovy`—LDAP or PAM, but not both. If you have configured both LDAP and PAM, MWS defaults to using LDAP.

If you need multiple authentication methods, you must add them to your local PAM configuration. See your distribution documentation for details.

- If you are configuring an MWS connection to your LDAP server, add the following parameters to `/opt/mws/etc/mws-config.groovy`:

```
ldap.server = "192.168.0.5"
ldap.port = 389
ldap.baseDNs = ["dc=acme,dc=com"]
ldap.bindUser = "cn=Manager,dc=acme,dc=com"
ldap.password = "*****"
ldap.directory.type = "OpenLDAP Using InetOrgPerson Schema"
```

*This is just an example LDAP connection. Be sure to use the appropriate domain controllers (dc) and common names (cn) for your environment.*

**i** If you followed the Adaptive Computing tutorial, [Setting up OpenLDAP on CentOS 6 on page 71](#), your `ldap.directory.type` should be set to "OpenLDAP Using InetOrgPerson Schema." However, the use of other schemas is supported. For more information see [LDAP Configuration using mws-config.groovy on page 1402](#).

**i** To see how to configure a secure connection to the LDAP server, see [Securing the LDAP connection on page 1416](#).

- If you are configuring MWS to use PAM, add the the `pam.configuration.service` parameter to the `mws-config.groovy` file. For example:

```
pam.configuration.service = "login"
```

*This is just an example PAM configuration file name. Make sure you specify the name of the configuration file you want MWS to use.*

**i** For more information about PAM configuration with MWS, see [PAM \(pluggable authentication module\) configuration using mws-config.groovy on page 1405](#).

**⚠** There is a security risk when authenticating local users through your PAM configuration. This behavior is highly discouraged and not supported by Adaptive Computing.

- f. Add the `grails.mongo.username` and `grails.mongo.password` parameters to the `mws-config.groovy` file. Use the MWS credentials you added to MongoDB in the [Preparing for installation on page 17](#) section.

```
...  
grails.mongo.username = "mws_user"  
grails.mongo.password = "secret3"
```

- 8. Add the following lines to the end of `/etc/tomcat6/tomcat6.conf`:

```
CATALINA_OPTS="-DMWS_HOME=/opt/mws -Xms256m -Xmx3g -XX:MaxPermSize=384m -  
Dfile.encoding=UTF8"  
JAVA_HOME="/usr/java/latest"
```

**i** Some Linux distributions use `/etc/default/tomcat6` or `/etc/sysconfig/tomcat6` instead of `/etc/tomcat6/tomcat6.conf`.

- 9. Deploy the `mws.war` file and start Tomcat.

```
[root]# chkconfig tomcat6 on  
[root]# service tomcat6 stop  
[root]# cp /tmp/mws-install/mws-8.0.0/mws.war /usr/share/tomcat6/webapps  
[root]# service tomcat6 start
```

- 10. Navigate to `http://localhost:8080/mws/` in a web browser to verify that MWS is running (you will see some sample queries and a few other actions).
- 11. Log in to MWS to verify that your credentials are working. (Your login credentials are the `auth.defaultUser.username` and `auth.defaultUser.password` values you set in the `/opt/mws/etc/mws-config.groovy` file.)



**i** If you encounter problems, or if the application does not seem to be running, see the steps in [Moab Web Services issues](#) on page 82.

## Additional configuration

### Configuring SSL in Tomcat

To configure SSL in Tomcat, please refer to the Apache Tomcat [documentation](#).

## Related topics

## Setting up OpenLDAP on CentOS 6

These instructions are intended to help first-time LDAP administrators get up and running. The following procedures contain instructions for getting started using OpenLDAP on a CentOS 6 system. For more complete information on how to set up OpenLDAP see the [OpenLDAP documentation](#).

- [Installing and configuring OpenLDAP on Centos 6 on page 36](#)
- [Adding an organizational unit \(OU\) on page 40](#)
- [Adding a user on page 40](#)
- [Adding a group on page 41](#)
- [Adding a user to a group on page 41](#)

**i** Adaptive Computing is not responsible for creating, maintaining, or supporting customer LDAP or Active Directory configurations.

## Installing and configuring OpenLDAP on Centos 6

First, you will need to install OpenLDAP. These instructions explain how you can do this on a CentOS 6 system.

### To install and configure OpenLDAP on Centos 6

1. Run the following command:

```
[root]# yum -y install openldap openldap-clients openldap-servers
```

2. Generate a password hash to be used as the admin password. This password hash will be used when you create the root user for your LDAP installation. For example:

```
[root]# slappasswd
New password : p@ssw0rd
Re-enter new password : p@ssw0rd
{SSHA}5lPFVw19zeh7LT53hQH69znpj8TuBrLv
```

3. Add the root user and the root user's password hash to the OpenLDAP configuration in the `olcDatabase={2}bdb.ldif` file. The root user will have permissions to add other users, groups, organizational units, etc. Do the following:

- a. Run this command:

```
[root]# cd /etc/openldap/slapd.d/cn=config
[root]# vi olcDatabase=\{2\}bdb.ldif
```

- b. If the **olcRootPW** attribute does not already exist, create it. Then set the value to be the hash you created from `slappasswd`. For example:

```
olcRootPW: {SSHA}51PFVw19zeh7LT53hQH69znzj8TuBrLv
...
```

4. While editing this file, change the distinguished name (DN) of the **olcSuffix** to something appropriate. The suffix typically corresponds to your DNS domain name, and it will be appended to the DN of every other LDAP entry in your LDAP tree.

For example, let's say your company is called Acme Corporation, and that your domain name is "acme.com." You might make the following changes to the `olcDatabase={2}bdb.ldif` file:

```
olcSuffix: dc=acme,dc=com
...
olcRootDN: cn=Manager,dc=acme,dc=com
...
olcRootPW: {SSHA}51PFVw19zeh7LT53hQH69znzj8TuBrLv
...
```

 Throughout the following examples in this topic, you will see `dc=acme,dc=com`. "acme" is only used as an example to illustrate what you would use as your own domain controller if your domain name was "acme.com." You should replace any references to "acme" with your own organization's domain name.

 Do not set the cn of your root user to "root" (`cn=root,dc=acme,dc=com`), or OpenLDAP will have problems.

5. Modify the DN of the root user in the `olcDatabase={1}monitor.ldif` file to match the **olcRootDN** line in the `olcDatabase={2}bdb.ldif` file. Do the following:

- a. Run this command to edit the `olcDatabase={2}bdb.ldif` file:

```
[root]# vi olcDatabase=\{1\}monitor.ldif
```

- b. Modify the **olcAccess** line so that the **dn.base** matches the **olcRootDN** from the `olcDatabase={2}bdb.ldif` file. (In this example, **dn.base** should be "cn=Manager,dc=acme,dc=com".)

```
olcAccess: {0}to * by
dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read by
dn.base="cn=Manager,dc=acme,dc=com" read by * none
```

- c. Now the root user for your LDAP is `cn=Manager,dc=acme,dc=com`. The root user's password is the password that you entered using `slappasswd` (see step 2), which, in this example, is **p@ssw0rd**
6. Hide the password hashes from users who should not have permission to view them.

 A full discussion on configuring access control in OpenLDAP is beyond the scope of this tutorial. For help, see [the OpenLDAP Access Control documentation](#).

- a. Run this command to edit the `olcDatabase\=\{2\}bdb.ldif` file:

```
[root]# vi olcDatabase\=\{2\}bdb.ldif
```

- b. Add the following two lines to the end of the file to restrict users from viewing other users' password hashes.

```
olcAccess: {0}to attrs=userPassword by self write by
dn.base="cn=Manager,dc=acme,dc=com" write by anonymous auth by * none
olcAccess: {1}to * by dn.base="cn=Manager,dc=acme,dc=com" write by self write by
* read
```

*These lines allow a user to read and write his or her own password. It also allows a manager to read and write anyone's password. Anyone, including anonymous users, is allowed to view non-password attributes of other users.*

7. Make sure that OpenLDAP is configured to start when the machine starts up, and start the OpenLDAP service.

```
[root]# chkconfig slapd on
[root]# service slapd start
```

8. Now, you must manually create the "dc=acme,dc=com" LDAP entry in your LDAP tree.

An LDAP directory is analogous to a tree. Nodes in this tree are called LDAP "entries" and may represent users, groups, organizational units, domain controllers, or other objects. The attributes in each entry are determined by the LDAP schema. In this tutorial we will build entries based on the `InetOrgPerson` schema (which ships with OpenLDAP by default).

In order to build our LDAP tree we must first create the root entry. Root entries are usually a special type of entry called a domain controller (DC). Because we are assuming that the organization is called Acme Corporation, and that the domain is "acme.com," we will create a domain controller LDAP entry called `dc=acme,dc=com`. Again, you will need to replace "acme" with your organization's domain name. Also note that `dc=acme,dc=com` is what is called an LDAP distinguished name (DN). An LDAP distinguished name uniquely identifies an LDAP entry.

Do the following:

- a. Create a file called `acme.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi acme.ldif
```

- b. Add the following lines in `acme.ldif`:

```
dn: dc=acme,dc=com
objectClass: dcObject
objectClass: organization
dc: acme
o : acme
```

- c. Now add the contents of this file to LDAP. Run this command:

```
[root]# ldapadd -f acme.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

- d. Verify that your entry was added correctly.

```
[root]# ldapsearch -x -LLL -b dc=acme,dc=com
dn: dc=acme,dc=com
objectClass: dcObject
objectClass: organization
dc: acme
o: acme
```

9. Run the following:

```
[root]# sudo iptables -L
[root]# sudo service iptables save
```

10. By default, the CentOS 6 firewall will block external requests to OpenLDAP. In order to allow MWS to access LDAP, you will have to configure your firewall to allow connections on port 389. (Port 389 is the default LDAP port.)

Configuring your firewall is beyond the scope of this tutorial; however, it may be helpful to know that the default firewall on CentOS is a service called `iptables`. (For more information, see the documentation on [iptables](#).) In the most basic case, you may be able to add a rule to your firewall that accepts connections to port 389 by doing the following:

- a. Edit your `iptables` file:

```
[root]# vi /etc/sysconfig/iptables
```

- b. Add the following line *after* all the **ACCEPT** lines but *before* any of the **REJECT** lines in your `iptables` file:

```
# ... lines with ACCEPT should be above
-A INPUT -p tcp --dport 389 -j ACCEPT
# .. lines with REJECT should be below
```

For example, here is a sample `iptables` file with this line added:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -p tcp --dport 389 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

- c. Now reload `iptables`.

```
[root]# service iptables reload
```

**i** Although providing instructions is beyond the scope of this tutorial, it is also highly recommended that you set up OpenLDAP to use SSL or TLS security to prevent passwords and other sensitive data from being sent in plain text. For information on how to do this, see the [OpenLDAP TLS documentation](#).

Now that you have installed and set up Open LDAP, you are ready to add organizational units (see [Adding an organizational unit \(OU\) on page 40](#)).

## Adding an organizational unit (OU)

These instructions will describe how to populate the LDAP tree with organizational units (OUs), groups, and users, all of which are different types of LDAP entries. The examples that follow also presume an InetOrgPerson schema, because the InetOrgPerson schema is delivered with OpenLDAP by default.

### To add an organizational unit (OU) entry to the LDAP tree

In this example, we are going to add an OU called "Users."

1. Create a temporary file called `users.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi users.ldif
```

2. Add these lines to `users.ldif`:

```
dn: ou=Users,dc=acme,dc=com
objectClass: organizationalUnit
ou: Users
```

3. Add the contents of `users.ldif` file to LDAP.

```
[root]# ldapadd -f users.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a user

### To add a user to LDAP

In this example, we will add a user named "Bob Jones" to LDAP inside the "Users" OU.

1. Create a temporary file called `bob.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi bob.ldif
```

2. Add these lines to `bob.ldif`:

```
dn: cn=Bob Jones,ou=Users,dc=acme,dc=com
cn: Bob Jones
sn: Jones
objectClass: inetOrgPerson
userPassword: p@ssw0rd
uid: bjones
```

3. Add the contents of bob.ldif file to LDAP.

```
[root]# ldapadd -f bob.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a group

### To add a group to LDAP

In this example, we will add a group called "Engineering" to LDAP inside the "Users" OU.

1. Create a temporary file called engineering.ldif. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the /tmp folder.)

```
[root]# cd /tmp
[root]# vi engineering.ldif
```

2. Add these lines to engineering.ldif:

```
dn: cn=Engineering,ou=Users,dc=acme,dc=com
cn: Engineering
objectClass: groupOfNames
member: cn=Bob Jones,ou=Users,dc=acme,dc=com
```

3. Add the contents of engineering.ldif file to LDAP.

```
[root]# ldapadd -f engineering.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a user to a group

### To add a user to an LDAP group

In this example, we will add an LDAP member named "Al Smith" to the "Engineering" LDAP group. This example assumes that user, Al Smith, has already been added to LDAP.

**i** Before you add a user to an LDAP group, the user must first be added to LDAP. For more information, see [Adding a user on page 40](#).

1. Create a temporary file called addUserToGroup.ldif. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the /tmp folder.)

```
[root]# cd /tmp
[root]# vi addUserToGroup.ldif
```

2. Add these lines to addUserToGroup.ldif:

```
dn: cn=Engineering,ou=Users,dc=acme,dc=com
changetype: modify
add: member
member: cn=Al Smith,ou=Users,dc=acme,dc=com
```

3. Now add the contents of addUserToGroup.ldif file to LDAP.

```
[root]# ldapadd -f addUserToGroup.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

Related topics

## Moab Workload Manager configuration options

The following is a list of commonly used configure options. For a complete list, use `./configure --help` when configuring Moab.

Option	Description	Example
<b>--with-flexlm</b>	Causes Moab to install the <code>license.mon.flexLM.pl</code> script in the <code>/opt/moab/tools</code> directory. For more information about this script, see the <a href="#">Interfacing to FLEXlm on page 653</a> section in the Moab Administrator Guide.	<pre>[root]# ./configure --with-flexlm</pre>
<b>--with-homedir</b>	Specifies the location of the Moab configuration directory and the MOABHOMEDIR environment variable. The default location is <code>/opt/moab</code> .  <b>i</b> MOABHOMEDIR is automatically set on some distributions during installation, when the <code>--with-profile</code> option is enabled.	<pre>[root]# ./configure --with-homedir=/var/moab</pre> <i>The Moab HPC Suite home directory will be <code>/var/moab</code> instead of the default <code>/opt/moab</code>.</i>

Option	Description	Example
<p><b>--with-init</b></p>	<p>Enables the installation of a distribution-specific <code>/etc/init.d/moab</code> service startup script.</p> <p>This option is required if you want to install this script onto a new system. If you do not set this option, you must manually set up the Moab daemon service.</p> <p>The startup script is located at <code>OS/EL/etc/init.d/moab</code>.</p> <div data-bbox="506 688 857 968" style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> The TORQUE and Moab HPC Suite initialization scripts are provided in the <code>contrib/init.d</code> directory as a courtesy and may be modified at your discretion to work on your system.</p> </div>	<pre data-bbox="906 317 1432 363" style="border: 1px dashed gray; padding: 5px;">[root]# ./configure --with-init</pre>
<p><b>--prefix</b></p>	<p>Specifies the location of the binaries and libraries of the Moab install.</p> <p>The default location is <code>/opt/moab</code>.</p>	<pre data-bbox="906 1020 1432 1066" style="border: 1px dashed gray; padding: 5px;">[root]# ./configure --prefix=/usr/local</pre>
<p><b>--with-profile</b></p>	<p>Enables the installation of distribution-specific <code>/etc/profile.d/moab.[c]sh</code> setup script for bash and cshell.</p> <p>The <code>MOABHOMEDIR</code>, <code>PERL5LIB</code>, <code>PATH</code> and <code>MANPATH</code> environment variables are setup to specify where the new moab configuration, scripts, binaries and man pages reside. If you do not set this option, these scripts are not installed, and you must manually perform this set up.</p> <p>The environment setup scripts are located at <code>OS/EL/etc/profile.d/moab.[c]sh</code>.</p>	<pre data-bbox="906 1230 1432 1276" style="border: 1px dashed gray; padding: 5px;">[root]# ./configure --with-profile</pre>

Option	Description	Example
<b>--with-am</b>	<p>Specifies that you want to configure Moab with Moab Accounting Manager.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> There is a similar <code>--with-torque</code> option that configures Moab with TORQUE, but you do not need to specify this option if you install the "torque" tarball version.</p> </div>	<pre>[root]# ./configure --with-am</pre>

## Trusting servers in Java

### Prerequisites

Some of these instructions refer to `JAVA_HOME`, which must point to the same directory that Tomcat uses. To set `JAVA_HOME`, do this:

```
[root]# source /etc/tomcat6/tomcat6.conf
```

Your system administrator might have defined Tomcat's `JAVA_HOME` in a different file.

### Retrieve the server's X.509 public certificate

To retrieve the server's certificate, use the following command:

```
[root]# $JAVA_HOME/bin/keytool -printcert -rfc -sslserver <servername>:<port> > /tmp/public.cert.pem
```

Replace `<servername>` with the server's host name and `<port>` with the secure port number. The default port for https is 443. The default port for ldaps is 636. If successful, `/tmp/public.cert.pem` contains the server's public certificate. Otherwise, `/tmp/public.cert.pem` contains an error message. This message is typical: `keytool error: java.lang.Exception: No certificate from the SSL server`. This message suggests that the server name or port is incorrect. Consult your IT department to determine the correct server name and port.

### Add the server's certificate to Java's keystore

Java stores trusted certificates in a database known as the keystore. Because each new version of Java has its own keystore, you need to add the server certificate to the Java keystore (using the steps below) every time you install a new version of Java.

Java's keystore is located at `$JAVA_HOME/lib/security/cacerts`. If Tomcat's `JAVA_HOME` points to a JDK, then the keystore is located at `$JAVA_HOME/jre/lib/security/cacerts`. To add the server certificate to the keystore, run the following command:

```
[root]# $JAVA_HOME/bin/keytool -import -trustcacerts -file /tmp/public.cert.pem -alias
<servername> -keystore $JAVA_HOME/lib/security/cacerts
```

You will be prompted for the keystore password, which is "changeit" by default.

**i** Your system administrator might have changed this password.

After you've entered the keystore password, you'll see the description of the server's certificate. At the end of the description it prompts you to trust the certificate.

```
Trust this certificate? [no]:
```

Type **yes** and press **Enter** to add the certificate to the keystore.

## Upgrading

### Upgrading MongoDB

Adaptive Computing strongly recommends upgrading MongoDB to version 2.4. Support for environments using MongoDB 2.0 is now deprecated and will be removed in future releases. Please refer to [docs.mongodb.org](https://docs.mongodb.org) for instructions on how to upgrade MongoDB. Note that you must pay close attention to the information regarding instances with `auth` enabled (as this is the recommended setup for Moab HPC Suite).

```
[root]# service mongod stop
[root]# yum remove mongo20-10gen-server mongo20-10gen
[root]# yum install mongo-10gen-server --exclude mongodb-org,mongodb-org-server
[root]# service mongod start
```

**i** Note that the settings in the `/etc/mongod.conf` file were saved in `/etc/mongod.conf.rpmsave` while removing MongoDB 2.0. You may need to be restore any custom settings after MongoDB 2.4 is installed in the new `/etc/mongod.conf` file (for example, "auth = true").

After upgrading to MongoDB 2.4, you should verify that the MongoDB credentials were preserved:

```
[root]# mongo -u mws_user mws -p
MongoDB shell version: 2.4.8
connecting to: mws
> show collections
event
mongeez
permission
...
```

### Upgrading TORQUE

TORQUE 5.0.0 is not backward compatible with versions of TORQUE prior to 4.0. When you upgrade to TORQUE 5.0.0, all MOM and server daemons must be upgraded at the same time.

The job format is compatible between 5.0.0 and previous versions of TORQUE. Any queued jobs will upgrade to the new version with the exception of job arrays in TORQUE 2.4 and earlier. It is not recommended to upgrade TORQUE while jobs are in a running state.

## Job arrays

Job arrays from TORQUE version 2.5 and 3.0 are compatible with TORQUE 5.0.0. Job arrays were introduced in TORQUE version 2.4 but modified in 2.5. If upgrading from TORQUE 2.4, you need to make sure all job arrays are complete before upgrading.

## serverdb

The `pbs_server` configuration is saved in the file `$TORQUE_HOME/server_priv/serverdb`. When running TORQUE 4.1.0 or later for the first time, this file converts from a binary file to an XML-like format. This format can be used by TORQUE versions 2.5 and 3.0, but not earlier versions. Back up the `$TORQUE_HOME/server_priv/serverdb` file before moving to TORQUE 4.1.0 or later.

## Jobs

Before upgrading the system, all running jobs must complete. To prevent queued jobs from starting, nodes can be set to offline or all queues can be disabled. Once all running jobs are complete, the upgrade can be made. Remember to allow any job arrays in version 2.4 to complete before upgrading. Queued array jobs will be lost.

## Cray

For upgrading TORQUE to 5.0.0 on a Cray system, refer to the [Installation Notes for Moab and TORQUE for Cray on page 1236](#) in Appendix G of the Moab Workload Manager Administrator Guide.

## To upgrade TORQUE

1. Shut down TORQUE.

```
[root]# qterm
[root]# momctl -s

* If running TORQUE 4.6.0 or later *
[root]# trqauthd -d

*If running a version of TORQUE earlier than 4.6.0 *
[root]# ps -efw | grep trqauthd
root      1487      1  0 Dec18 ?          00:00:00 /usr/sbin/trqauthd
adaptive 4830  4374  0 15:07 pts/0    00:00:00 grep trqauthd

[root]# kill -9 1487
```

2. Back up your `server_priv` directory.

```
[root]# tar -cvf backup.tar.gz $TORQUE_HOME/server_priv
```

3. If not already installed, install the Boost C++ headers.

```
[root]# yum install boost-devel
```

**i** For SLES, use `zypper install <package names>` instead of `yum install <package names>`.

#### 4. Install the latest TORQUE tarball.

```
[root]# cd /tmp
[root]# tar xzvf torque-5.0.0-<build number>.tar.gz
[root]# cd torque-5.0.0-<build number>
[root]# ./configure
[root]# make
[root]# make install
```

#### 5. If not already done, configure `pbs_server` and `pbs_mom` to start automatically at system boot.

```
* If Debian distribution, do the following *
[root]# cp contrib/init.d/debian.pbs_server /etc/init.d/pbs_server
[root]# cp contrib/init.d/debian.pbs_mom /etc/init.d/pbs_mom
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom

* If SLES distribution, do the following *
[root]# cp contrib/init.d/suse.pbs_server /etc/init.d/pbs_server
[root]# cp contrib/init.d/suse.pbs_mom /etc/init.d/pbs_mom
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom

* If RHEL distribution, do the following *
[root]# cp contrib/init.d/pbs_server contrib/init.d/pbs_mom /etc/init.d
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom
```

#### 6. Start the services.

```
[root]# service trqauthd start
[root]# service pbs_mom start
[root]# service pbs_server start
```

#### 7. Check the status of jobs in the queue and perform other checks for errors.

```
[root]# qstat
[root]# grep -i error /var/spool/torque/server_logs/*
[root]# grep -i error /var/spool/torque/mom_logs/*
```

## Upgrading Moab Workload Manager

The following instructions will guide you through a 6.1.x, 7.0.x, 7.1.x, 7.2.x, or 7.5.0 to 8.0.0 upgrade. Depending on which version of Moab you are presently running, upgrade instructions may vary, so unless otherwise noted, all instructions assume use of a RHEL operating system; notes for SLES users are added in appropriate places.

You might want to [test](#) the newest version of Moab on your system (before making the new version live) to verify your policies, scripts, and queues work the way you want them to.

If you are also upgrading TORQUE from an older version (pre-4.0), you may encounter a problem where Moab HPC Suite core files are regularly created in `/opt/moab`. This can be caused by old TORQUE library files used by Moab that try to authorize with the old TORQUE `pbs_iff` authorization daemon. You can resolve the problem by removing the old version library files from `/usr/local/lib`.

**i** Because many system-level files and directories are accessed during the installation, the instructions in this guide should be executed with root privileges.

You will see that the instructions execute commands as the root user. Please note that the same commands will work for a non-root user with the `sudo` command.

## To upgrade Moab

1. Untar the distribution file. For example:

```
[root]# tar -xzvf moab-<version>-<OS>.tar.gz
```

2. Change directory into the extracted directory.
3. Verify `/etc/yum.repos.d/epel.repo` exists and has the following lines. If not, create it and add these lines.

```
[epel]
name=Extra Packages for Enterprise Linux 6 - x86_64
mirrorlist=http://mirrors.fedoraproject.org/mirrorlist?repo=epel-6&arch=x86_64
failovermethod=priority
enabled=1
gpgcheck=1
gpgkey=http://download.fedoraproject.org/pub/epel/RPM-GPG-KEY-EPEL-6
```

**i** SLES users must add a repository to YaST. The URL of the repository is [http://download.opensuse.org/repositories/server:/database/SLE\\_11\\_SP2/](http://download.opensuse.org/repositories/server:/database/SLE_11_SP2/).

4. Configure the installation package.

Use the same configure options as when Moab was installed previously. If you cannot remember which options were used previously, check the `config.log` file in the directory where the previous version of Moab was installed from.

For a complete list of configure options, use `./configure --help`.

5. Stop Moab.

```
[root]# mschedctl -k
moab will be shutdown immediately
```

**i** While Moab is down, all currently running jobs continue to run on the nodes, the job queue remains intact, and new jobs cannot be submitted to Moab.

6. Before proceeding to the following steps, consider backing up your Moab Workload Manager home directory (`/opt/moab/` by default).

- If you use ODBC, you must upgrade to the 8.0.0 schema. See [Migrating Your Database to Newer Versions of Moab on page 808](#) for more information.
- Run the `make perldeps` command to install the necessary perl modules using CPAN (and install CPAN if you have not already done so). When first running CPAN, you will be asked for configuration information. It is recommended that you choose an automatic configuration. You will be prompted to provide input during module installation; running the `make perldeps` command with a script is not recommended.

```
[root]# yum install cpan
[root]# make perldeps
```

- Install Moab.

```
[root]# make install
```

**i** Default configuration files are installed during `make install`. Existing configuration files are not overwritten and the new files are given a `.dist` extension.

- Verify the version number is correct before starting the new server version.

```
[root]# moab --about

Defaults:  server=:42559  cfgdir=/opt/moab  vardir=/opt/moab
Build dir:  /tmp/develop
Build host: crom
Build date: Mon Jun 16 16:00:00 MST 2014
Build args: NA
Compiler Flags:  -D_M64 -D_BUILDDATETIME="2014061616" -DMUSEWEBSERVICES -
DMUSEZEROMQ -DMUSEMONGODB -DMMAX_GRES=512 -DMMAX_RANGE=2048 -DMMAX_TASK=32768 -fPIC
-gdwarf-3 -Wall -Wextra -DVALGRIND -x c++ -std=c++11 -DDMAX_PJOB=512 -D_GNU_SOURCE
Compiled as little endian.
Version: moab server master (revision 2014061616, changeset
90ce9f804ddd09b061238e438ecb4d117cc83e81)
Copyright (C) 2000-2014 by Adaptive Computing Enterprises, Inc. All Rights
Reserved.
```

- If you are upgrading from Moab Workload Manager 7.5 or earlier and use Moab Accounting Manager with the native interface, (`AMCFG[...] TYPE=native`), locate the entries in `moab.cfg` with the following form:

```
AMCFG[mam] *URL=exec:/// $HOME/tools/mam/*.*.mam.pl
```

Replace the matching entries with the following:

```
AMCFG[mam] CREATEURL=exec://$TOOLSDIR/mam/usage.quote.mam.pl
AMCFG[mam] STARTURL=exec://$TOOLSDIR/mam/usage.reserve.mam.pl
AMCFG[mam] PAUSEURL=exec://$TOOLSDIR/mam/usage.charge.mam.pl
AMCFG[mam] UPDATEURL=exec://$TOOLSDIR/mam/usage.charge.mam.pl
AMCFG[mam] RESUMEURL=exec://$TOOLSDIR/mam/usage.reserve.mam.pl
AMCFG[mam] ENDURL=exec://$TOOLSDIR/mam/usage.charge.mam.pl
AMCFG[mam] DELETEURL=exec://$TOOLSDIR/mam/lien.delete.mam.pl
```

- Start Moab.

```
[root]# moabd
```

- If you will be using Moab Web Services, you must configure a secret key. See [Securing communication using secret keys on page 30](#).

## Upgrading MWS

### Context

Before upgrading MWS, we recommend you upgrade to Java 7 and MongoDB 2.4. To upgrade Java, repeat the [Java installation instructions](#). To upgrade MongoDB, see [Upgrading MongoDB on page 45](#).



It is highly recommended that you perform a full database backup before updating your database. This can be done using the `mongodump` utility documented in the [MongoDB documentation](#).

### To perform an MWS upgrade

- Extract the contents of the MWS download tarball into a temporary directory. For example:

```
[root]# mkdir /tmp/mws-install
[root]# cd /tmp/mws-install
[root]# tar xvzf mws-8.0.0.tar.gz
[root]# cd /tmp/mws-install/mws-8.0.0
```

- Stop Tomcat, re-deploy `mws.war`, and remove the exploded `mws` directory.

```
# CentOS 6 example

[root]# service tomcat6 stop
[root]# cp /tmp/mws-install/mws-8.0.0/mws.war /usr/share/tomcat6/webapps
[root]# rm -rf /usr/share/tomcat6/webapps/mws
```

- Create the MWS home directory and subdirectories (for more information, see the "[Configuration on page 1772](#)" section of the *Moab Web Services Reference Guide*).



The default location for the MWS home directory is `/opt/mws`. These instructions assume the default location.

Here is a sample script for this setup:

```
[root]# mkdir -p \  
  /opt/mws/etc/mws.d \  
  /opt/mws/hooks \  
  /opt/mws/log \  
  /opt/mws/plugins \  
  /opt/mws/spool/hooks \  
  /opt/mws/utils  
[root]# chown -R tomcat:tomcat /opt/mws # Depending on your OS, the Tomcat username  
might be tomcat6.  
[root]# chmod -R 555 /opt/mws  
[root]# chmod u+w \  
  /opt/mws/log \  
  /opt/mws/plugins \  
  /opt/mws/spool \  
  /opt/mws/spool/hooks \  
  /opt/mws/utils
```

4. Copy the extracted utility files to the utility directory created above and give the tomcat user ownership of the directory.

```
[root]# cd /tmp/mws-install/mws-8.0.0/utils  
[root]# cp * /opt/mws/utils  
[root]# chown tomcat:tomcat /opt/mws/utils/*
```

5. Set up the MWS configuration files. In the extracted directory are several configuration files.
  - a. Merge the `/opt/mws/etc/mws-config.groovy.rpmnew` file with the old `/opt/mws/etc/mws-config.groovy` file by editing `/opt/mws/etc/mws-config.groovy`. (Note the addition of the "auditAppender" in the default logging configuration of `/opt/mws/etc/mws-config.groovy.rpmnew`.)



**i** If necessary, open port 5570 in the firewall.

Note that the **mws.suite** parameter and the **mam.\*** parameters have been moved to a suite-specific file in `/opt/mws/etc/mws.d/` and do not need to exist in `/opt/mws/etc/mws-config.groovy`.

Also note the new **\*messageQueue** parameters in `/opt/mws/etc/mws-config.groovy.rpmnew`. These are required and the value for **moab.messageQueue.secretKey** should match the value located in `/opt/moab/etc/moab-private.cfg`. If you have not yet configured a secret key, see [Securing communication using secret keys on page 30](#).

- b. Copy the `mws-config-hpc.groovy` file to `/opt/mws/etc/mws.d`.
  - c. Give the Tomcat user read access to `/opt/mws/etc/mws-config.groovy` and `/opt/mws/etc/mws.d/mws-config-*.groovy`.
6. Upgrade the schema of the `mws` database in MongoDB.

**!** You *must* perform this step, regardless of whether you upgraded MongoDB to version 2.4 or not.

Run the database migration script provided with MWS. (It is safe to run this script more than once. If for any reason, errors occur during the execution of the script, run it again.)

```
[root]# mongo -u mws_user mws /opt/mws/utils/db-migrate.js -p
```

**i** The script might take several minutes to execute.

7. Start Tomcat.

```
[root]# service tomcat6 start
```

8. Visit <http://localhost:8080/mws/> in a web browser to verify that MWS is running again.

You will see some sample queries and a few other actions.

9. Log into MWS to verify configuration. (The credentials are the values of **auth.defaultUser.username** and **auth.defaultUser.password** set in `/opt/mws/etc/mws-config.groovy`.)

**i** If you encounter problems, or if MWS does not seem to be running, see the steps in [Troubleshooting on page 78](#).

# RPM installation

## Installing Moab HPC Suite – Basic Edition

**i** The RPM installation only supports installation on Red Hat 6.5, CentOS 6.5 or Scientific Linux 6.5. Use the [Manual installation](#) instructions if installing on other supported operating systems.

**i** Because many system-level files and directories are accessed during the installation, the instructions in this guide should be executed with root privileges.

You will see that the instructions execute commands as the root user. Please note that the same commands will work for a non-root user with the `sudo` command.

## Dependencies and packages installation

### Install Java

Install the *Linux x64 RPM* version of Oracle® Java® 7 Runtime Environment.

**i** Oracle Java 7 Runtime Environment is the recommended Java environment, but Oracle Java 6 is also supported. All other versions of Java, including OpenJDK/IcedTea, GNU Compiler for Java, and so on cannot run Moab Web Services.

Do the following:

1. Download the *Linux x64 RPM* version of Oracle Java SE 7 JRE. (Go to the [Oracle Java 7 download page](#), copy the URL to the *Linux x64 RPM* version, then run the following command.)

```
[root]# wget <URL> -O jre-7-linux-x64.rpm
```

To verify that the download was successful, run the following on the RPM before installation:

```
[root]# rpm -qip jre-7-linux-x64.rpm
```

2. Run the following to install Java 7:

```
[root]# rpm -Uh jre-7-linux-x64.rpm
```

### Opening ports

A few ports need to be available through your firewall so components of the suite can communicate with each other. Some features of some components might need additional ports configured. The individual component documentation indicates when additional ports are needed.

- 8080: Needed for Moab Web Services web portal (http)

## To open ports in your firewall

- Use **iptables** for Red Hat-based distributions:

```
[[root]# iptables-save > /tmp/iptables.mod
[root]# vi /tmp/iptables.mod

# Add the following lines immediately *before* the line matching
# "-A INPUT -j REJECT --reject-with icmp-host-prohibited"
-A INPUT -p tcp --dport 8080 -j ACCEPT

[root]# iptables-restore < /tmp/iptables.mod
[root]# service iptables save
```

- Use **SuSEfirewall12** for SuSE-based distributions:

```
[root]# vi /etc/sysconfig/SuSEfirewall12

FW_SERVICES_EXT_TCP="8080"

[root]# service SuSEfirewall12_setup restart
```

## Installing the RPM suite

**i** If you want to build a custom RPM for component documentation (Moab Workload Manager, Moab Web Services or Moab Accounting Manager, refer to the Manual Installation Guide and follow instructions in the respective component.

### To install the RPM suite

1. If you are upgrading from a previous installation of Moab, back up your `/opt/moab/tools` directory to prevent losing modifications made to the perl scripts. If you are performing a clean installation of Moab HPC Suite, skip this step.

```
[root]# tar czf backup-tools.tar.gz /opt/moab/tools
```

2. Download the latest 8.0.0 RPM suite tarball (`moab-hpc-basic-suite-<version>-<timestamp>-<OS>.tar.gz`, for example) from the [Adaptive Computing](#) website.
3. Untar the downloaded package.

```
[root]# tar xzf moab-hpc-basic-suite-<version>-<timestamp>-<OS>.tar.gz
```

4. Change directories into the untarred directory.

**i** Consider reviewing the README file for additional details on using the RPM distribution tarball.

5. Install the suite repositories. The `-y` option installs with the default settings for the RPM suite.

**i** For a description of the options of the repository installer script, run:

```
[root]# ./install-rpm-repos.sh -h
```

```
[root]# ./install-rpm-repos.sh [<repository-directory>] [-y]
```

**i** If the installation returns the following warning line:

```
Warning: RPMDB altered outside of yum.
```

This is normal and can safely be ignored.

The [*<repository-directory>*] option is the directory where you want to copy the RPMs. If no argument is given, run "install-rpm-repos.sh -h" and note the default directory location. If the [*<repository-directory>*] already exists, RPMs will be added to the existing directory. No files are overwritten in [*<repository-directory>*]. A repository file is also created in `/etc/yum.repos.d/` and points to the [*<repository-directory>*] location.

For ease in repository maintenance, the install script fails if Adaptive Computing RPMs are copied to different directories. If a non-default [*<repository-directory>*] is specified, please use the same directory for future updates.

The script installs the `createrepo` package and its dependencies. You must answer "y" to all the questions in order for the RPM install of the suite to work. Additionally, the script installs the EPEL and 10gen repositories.

## 6. Test the repository:

```
[root]# yum search moab
```

If no error is given, the repository is correctly installed. The output will look similar to the following (varying slightly depending on the suite and build type):

```
...
moab-hpc-basic-suite.noarch : Moab HPC Basic Suite virtual package
moab-perl-RRDs.noarch : Moab RRDs
moab-tomcat-config.x86_64 : Tomcat Configuration for Moab Viewpoint and Web
Services
moab-verify-oracle-java.noarch : Java Validator for Moab Viewpoint and Web Services
moab-web-services.x86_64 : Moab Web Services
moab-workload-manager.x86_64 : Moab Workload Manager
moab-workload-manager-client.x86_64 : Moab Workload Manager Client
moab-workload-manager-common.x86_64 : Moab Workload Manager Common Files
moab-perl-data.noarch : Perl Configuration for perl packages by Adaptive Computing
moab-torque-client.x86_64 : TORQUE Client
moab-torque-common.x86_64 : TORQUE Common Files
moab-torque-devel.x86_64 : TORQUE Development Files
moab-torque-mom.x86_64 : TORQUE MOM agent
moab-torque-server.x86_64 : TORQUE Server
moab-web-services-hpc-configuration.x86_64 : MWS configuration for HPC
moab-workload-manager-hpc-configuration.x86_64 : MWM configuration for HPC
```

## 7. Install the suite package.

```
[root]# yum install moab-hpc-basic-suite
```

**i** If you encounter the following error:

```
...
--> Finished Dependency Resolution
krb5-workstation-1.6.1-62.el5.x86_64 from installed has depsolving problems
--> Missing Dependency: krb5-libs = 1.6.1-62.el5 is needed by package
krb5-workstation-1.6.1-62.el5.x86_64 (installed)
krb5-workstation-1.6.1-62.el5.x86_64 from installed has depsolving problems
--> Missing Dependency: krb5-libs = 1.6.1-62.el5 is needed by package
krb5-workstation-1.6.1-62.el5.x86_64 (installed)
Error: Missing Dependency: krb5-libs = 1.6.1-62.el5 is needed by package
krb5-workstation-1.6.1-62.el5.x86_64 (installed)
You could try using --skip-broken to work around the problem
You could try running: package-cleanup --problems
package-cleanup --dupes
rpm -Va --nofiles --nodigest
```

Install the `krb5-workstation` package, then execute the install suite package again.

```
[root]# yum install krb5-workstation
[root]# yum install moab-hpc-basic-suite
```

**i** If you encounter CURL library errors, make sure you are installing the correct version for your OS.

## 8. Install and prepare the MongoDB database by doing the following:

- a. Install `mongo-10gen-server`.

```
[root]# yum install mongo-10gen-server --exclude mongodb-org,mongodb-org-server
```

- b. Start MongoDB.

**RHEL and CentOS, and Scientific Linux:**

```
[root]# chkconfig mongod on
[root]# service mongod start

[ OK ]
```

**i** If using versions of MongoDB older than 2.4.8 you might see connection errors when trying to execute the Mongo shell, such as "Error: couldn't connect to server 127.0.0.1:27017 at src/mongo/shell/mongo.js:145".

- c. Add the required MongoDB users.

**i** The passwords used below (`secret1`, `secret2`, and `secret3`) are examples. Choose your own passwords for these users.

```
[root]# mongo
> use admin;
> db.addUser("admin_user", "secret1");
> db.auth("admin_user", "secret1");

> use moab;
> db.addUser("moab_user", "secret2");
> db.addUser("mws_user", "secret3", true);

> use mws;
> db.addUser("mws_user", "secret3");
> exit
```

**i** Because the `admin_user` has read and write permissions to the `admin` database, it also has read and write permissions to all other databases. See [Control Access to MongoDB Instances with Authentication](#) for more information.

- d. Enable authentication in MongoDB.

#### **RHEL and CentOS, and Scientific Linux:**

```
[root]# vi /etc/mongod.conf

auth = true

[root]# service mongod restart
```

## Installing PostgreSQL

If you plan to use Moab Workload Manager with ODBC, you must install a PostgreSQL database.

### To install PostgreSQL

1. Install and initialize PostgreSQL.

#### **CentOS, RHEL, and Scientific Linux**

```
[root]# yum install postgresql-server
[root]# service postgresql initdb
```

#### **SLES**

```
[root]# zypper install postgresql-server
[root]# service postgresql start
```

2. Configure trusted connections.

**i** Edit or add a "host" line in the `pg_hba.conf` file for the interface from which the server(s) (for example, Moab Workload Manager and/or Moab Accounting Manager) will be connecting to the database and ensure that it specifies a secure password-based authentication method (for example, md5).

```
[root]# vi /var/lib/pgsql/data/pg_hba.conf
# IPv4 local connections:
host    all             all             127.0.0.1/32      md5
# IPv6 local connections:
host    all             all             ::1/128           md5
```

### 3. Configure PostgreSQL to accept connections from your host.

```
[root]# vi /var/lib/pgsql/data/postgresql.conf
# Uncomment the listen addresses line in the configuration:
listen_addresses = 'localhost'          # what IP address(es) to listen on;
```

### 4. Start or restart the database.

```
[root]# chkconfig postgresql on
[root]# service postgresql restart
```

#### Related topics

- [Welcome on page xxiii](#)

## Configuration

### Configuring TORQUE

These instructions describe how to configure and start TORQUE.

## Prerequisites

- TORQUE requires certain ports to be open for essential communication:
  - For client communication to `pbs_server`, all privileged ports must be open (ports under 1024).
  - For `pbs_server` communication to `pbs_mom`, the default port is 15003.
  - For `pbs_mom` to `pbs_server`, the default port is 15001.

For more information on how to configure the ports that TORQUE uses for communication, see [Configuring ports on page 2555](#).

**i Important:** If you intend to use TORQUE 5.0.0 with Moab, you must run Moab version 8.0.0 or later. TORQUE 5.0.0 will not work with versions earlier than Moab 8.0.0.

- Make sure your host (with the correct IP address) is in your `/etc/hosts` file.

## To configure TORQUE

1. Add nodes to the `/var/spool/torque/server_priv/nodes` file. For information on syntax and options for specifying compute nodes, see [Managing nodes on page 2617](#).
2. Start the servers.

```
[root]# service trqauthd start
[root]# service pbs_server start
[root]# service pbs_mom start
```

## To configure MOMs

1. To set up the MOMs, in the directory of the unpackaged tarball, copy the `torque-common` and `torque-mom` RPM files to each MOM node.

```
[root]# scp RPMs/moab-torque-common-*.rpm <mom-node>
[root]# scp RPMs/moab-torque-mom-*.rpm <mom-node>
```

2. On each MOM node, install the RPMs, making sure that `torque-mom-common` is installed first.

```
[root]# ssh root@<mom-node>
[root]# yum install moab-torque-common-*.rpm moab-torque-mom-*.rpm
```

3. By default, on each MOM node, the `/var/spool/torque/server_name` file contains the hostname of the current host. If `pbs_server` is running on another host, change the name.

```
[root]# echo <pbs_server's_hostname> > /var/spool/torque/server_name
```

4. Edit the `/var/spool/torque/mom_priv/config` file on each node. This file is identical for all compute nodes and can be created on the head node and distributed in parallel to all systems.

```
[root]# vi /var/spool/torque/mom_priv/config
$pbsserver      headnode      # hostname running pbs server
$logevent       225           # bitmap of which events to log
```

### Related topics

- [Installing Moab HPC Suite - Basic Edition on page 54](#)
- [Configuring Moab Workload Manager on page 61](#)
- [Component documentation on page 86](#)

## Configuring Moab Workload Manager

These instructions describe how to configure and start Moab Workload Manager (Moab).

### To configure Moab Workload Manager

1. Source the following file to add the Moab executable directories to your current shell `$PATH` environment.

```
[root]# . /etc/profile.d/moab.sh
```

2. Copy your license file into the same directory as `moab.cfg` (`/opt/moab/etc/` by default). For example:

```
[root]# cp moab.lic $MOABHOMEDIR/etc/moab.lic
```

To verify the current status of your license, use `moab --about`.

Moab checks the status of the license every day just after midnight. At 60 and 45 days before, and daily from 30 days before license expiration to and including the license expiration date, Moab sends an e-mail to all level 1 administrators informing them of the pending Moab license expiration. A log record is also made of the upcoming expiration event. For the notifications to occur correctly, you must enable administrator email notification (see "[Notifying Administrators of Failures on page 684](#)" in the Moab Workload Manager Administrator Guide) and `moab.cfg` must contain email addresses for level 1 administrators. For example:

```
ADMINCFG[1]  USERS=u1,u2,u3[,...]
USERCFG[u1]  EMAILADDRESS=u1@company.com
USERCFG[u2]  EMAILADDRESS=u2@company.com
USERCFG[u3]  EMAILADDRESS=u3@company.com
MAILPROGRAM DEFAULT
```



Moab will not run without a license. For information about obtaining a trial license, please contact [Adaptive Computing](#).

3. Start Moab.

```
[root]# chkconfig moab on
[root]# service moab start
```

- If you have a resource manager configured, submit a sleep job as a non-root user and verify the job is running.

**i** If you do not have a resource manager configured, skip this step. For TORQUE, you can configure a basic queue.

```
qmgr -c "set server scheduling=true"
qmgr -c "create queue batch queue_type=execution"
qmgr -c "set queue batch started=true"
qmgr -c "set queue batch enabled=true"
qmgr -c "set queue batch resources_default.nodes=1"
qmgr -c "set queue batch resources_default.walltime=3600"
qmgr -c "set server default_queue=batch"
```

```
[root]# su - moab
[moab]$ echo sleep 150 | msub
[moab]$ showq
[moab]$ exit
```

## 5. Connecting Moab to MongoDB

If you will be installing Moab Web Services, connect Moab to MongoDB using the following instructions:

**i** The `USEDATABASE` parameter is unrelated to the MongoDB configuration.

- In `/opt/moab/etc/moab.cfg`, set the **MONGOSERVER** parameter to the correct location of the MongoDB server. By default, Moab assumes it is on the same server.

```
MONGOSERVER <host>[:<port>]
```

- In the `/opt/moab/etc/moab-private.cfg` file, set the **MONGOUSER** and **MONGOPASSWORD** parameters to the MongoDB `moab_user` credentials you set.

```
MONGOUSER      moab_user
MONGOPASSWORD  secret2
```

- Verify that Moab is able to connect to MongoDB.

```
[root]# service moab restart
[root]# mdiag -S
...
Mongo connection (localhost) is up (credentials are set)
...
```

### Related topics

- [Installing Moab HPC Suite - Basic Edition on page 54](#)
- [Configuring TORQUE on page 59](#)
- [Component documentation on page 86](#)

## Configuring Moab Web Services

These instructions describe how to configure Moab Web Services (MWS).

## To configure Moab Web Services

### 1. Start Moab.

```
[root]# service moab start
```

### 2. Set up the MWS configuration file.

#### a. In the `/opt/mws/etc/mws-config.groovy` file, change these settings:

- **auth.defaultUser.username:** Any value you like, or leave as is.
- **auth.defaultUser.password:** Any value you like, but choose a strong password.

```
[root]# vi /opt/mws/etc/mws-config.groovy

// Change these to be whatever you like.
auth.defaultUser.username = "moab-admin"
auth.defaultUser.password = "changeme!"
```



If you do not change **auth.defaultUser.password**, your MWS will not be secure (because anyone reading these instructions would be able to log into your MWS). Here are some [tips](#) for choosing a good password.

#### b. Do *one* of the following:

- If you are configuring an MWS connection to your LDAP server, add the following parameters:

```
ldap.server = "192.168.0.5"
ldap.port = 389
ldap.baseDNs = ["dc=acme,dc=com"]
ldap.bindUser = "cn=Manager,dc=acme,dc=com"
ldap.password = "*****"
ldap.directory.type = "OpenLDAP Using InetOrgPerson Schema"
```

*This is just an example LDAP connection. Be sure to use the appropriate domain controllers (dc) and common names (cn) for your environment.*



If you followed the Adaptive Computing tutorial, [Setting up OpenLDAP on CentOS 6 on page 71](#), your **ldap.directory.type** should be set to "OpenLDAP Using InetOrgPerson Schema." However, the use of other schemas is supported. For more information see [LDAP Configuration using mws-config.groovy on page 1402](#).



To see how to configure a secure connection to the LDAP server, see [Securing the LDAP connection on page 1416](#).

- If you are configuring MWS to use PAM, add the the **pam.configuration.service** parameter to the `mws-config.groovy` file. For example:

```
pam.configuration.service = "login"
```

*This is just an example PAM configuration file name. Make sure you specify the name of the configuration file you want MWS to use.*

**i** For more information about PAM configuration with MWS, see [PAM \(pluggable authentication module\) configuration using mws-config.groovy](#) on page 1405.

**!** There is a security risk when authenticating local users through your PAM configuration. This behavior is highly discouraged and not supported by Adaptive Computing.

**i** You can configure only one authentication method in `mws-config.groovy`—LDAP or PAM, but not both. If you have configured both LDAP and PAM, MWS defaults to using LDAP. If you need multiple authentication methods, you must add them to your local PAM configuration. See your distribution documentation for details.

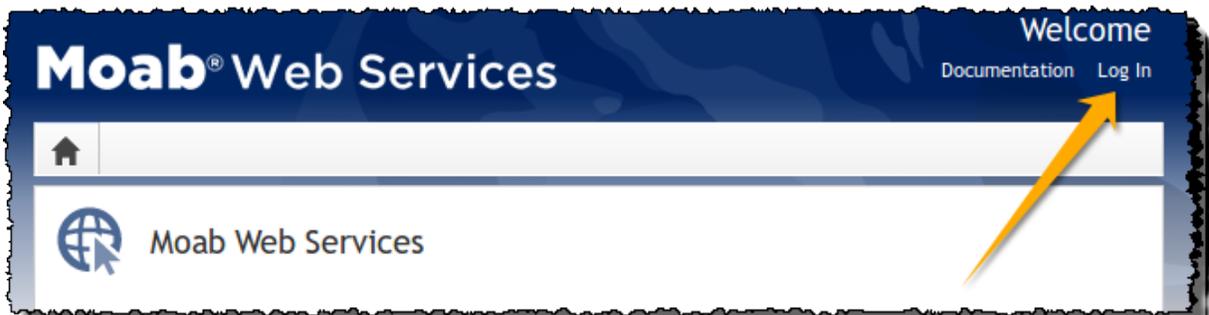
- c. Add the `grails.mongo.username` and `grails.mongo.password` parameters to the `mws-config.groovy` file. Use the MWS credentials you added to MongoDB in the [Installing Moab HPC Suite - Basic Edition](#) on page 54 section.

```
...
grails.mongo.username = "mws_user"
grails.mongo.password = "secret3"
```

3. Start or restart Tomcat.

```
[root]# chkconfig tomcat6 on
[root]# service tomcat6 restart
```

4. Navigate to `http://localhost:8080/mws/` in a web browser to verify that MWS is running (you will see some sample queries and a few other actions).
5. Log in to MWS to verify that your credentials are working. (Your login credentials are the `auth.defaultUser.username` and `auth.defaultUser.password` values you set in the `/opt/mws/etc/mws-config.groovy` file.)



**i** If you encounter problems, or if the application does not seem to be running, see the steps in [Moab Web Services issues on page 82](#)

## Installing Insight

### Context

Because Insight accumulates data for one cluster at a time, one Insight daemon should service one Moab instance. However, you can configure PostgreSQL to aggregate data using database replication mechanisms if you want cross-cluster .

**i** Moab Workload Manager and Insight both tend to heavily consume system resources. Therefore, we strongly recommend that the Insight daemon and the Moab Workload Manager server run on separate servers. For these installation instructions, the "Moab server" refers to one server and the "Insight server" refers to another server.

### Requirements

- Software:
  - 64-bit Linux server
  - RedHat or CentOS version 6.5 or greater
  - Sun® Java® 7 JRE
  - Postgres 9.3
- Hardware:
  - At least 8 cores
  - 8 GB of RAM
  - A dedicated 1 Gbit channel between Insight and Moab
  - 128 GB local SSD for swap
  - At least 512 GB disk

### To install Insight

1. Install Moab 8.0.0 or later on the Moab server.
2. Install Java 7 JRE on the Insight server. See [Install Java on page 54](#) for more information.
3. On the Insight server, download the latest 8.0.0 RPM suite tarball (moab-hpc-basic-suite-*<version>*-*<timestamp>*-*<OS>*.tar.gz, for example) from the [Adaptive Computing](#) website.
4. On the Insight server, untar the downloaded package.

```
[root]# tar xzf moab-hpc-basic-suite-<version>-<timestamp>-<OS>.tar.gz
```

5. On the Insight server, change directories into the untarred directory.

6. On the Insight server, install the suite repositories. The `-y` option installs with the default settings for the RPM suite.

**i** For a description of the options of the repository installer script, run:

```
[root]# ./install-rpm-repos.sh -h
```

```
[root]# ./install-rpm-repos.sh [<repository-directory>] [-y]
```

**i** If the installation returns the following warning line:

```
Warning: RPMDB altered outside of yum.
```

This is normal and can safely be ignored.

The script installs the `createrepo` package and its dependencies. You must answer "y" to all the questions in order for the RPM install of the suite to work. Additionally, the script installs the EPEL and 10gen repositories.

7. On the Insight server, install the Insight RPM. This will also install PostgreSQL.

```
[root]# yum install moab-insight
```

**i** If the installation returns the following warning line:

```
warning: rpmts_HdrFromFdno: Header V4 RSA/SHA1 Signature, key ID
952741e1: NOKEY
```

```
Retrieving key from file:///opt/adaptive-rpm-repository/key/GPG_
ADAPTIVE_COMPUTING_INC_EL_6_KEY
```

```
Importing GPG key 0x952741E1:
```

```
Userid: "Adaptive Computing Enterprises, Inc. (EL 6 key)
<info@adaptivecomputing.com>"
```

```
From : /opt/adaptive-rpm-repository/key/GPG_ADAPTIVE_COMPUTING_INC_EL_
6_KEY
```

This is normal. You can safely input `y` and continue.

8. On the Insight server, set up the PostgreSQL database.

- a. Initialize PostgreSQL.

```
[root]# /etc/init.d/postgresql-9.3 initdb
```

- b. Enable TCP connections instead of locally authenticated connections.

```
[root]# sed -i '/^host/s/ident/md5/' /var/lib/pgsql/9.3/data/pg_hba.conf
```

- c. If you are also using MWS, enable remote TCP connections.

```
[root]# echo 'host all all 0.0.0.0/32 md5' >> /var/lib/pgsql/9.3/data/pg_hba.conf
```



If you only want to enable access on one network interface, use the network interface IP address instead of 0.0.0.0.

- d. Start PostgreSQL.

```
[root]# service postgresql-9.3 start
```

- e. Switch to a PostgreSQL super-user. You will run the remaining commands in this step as that user.

```
[root]# su - postgres
```

- f. Create the Insight user and databases.

```
psql
CREATE USER moab_insight WITH PASSWORD 'changeme!';
CREATE DATABASE moab_insight WITH OWNER=moab_insight;
CREATE DATABASE moab_insight_reference WITH OWNER=moab_insight;
\q
```

- g. Initialize the `moab_insight` database. This sets up event triggers so that database schema validation works correctly in Insight.

```
psql moab_insight -f /opt/insight/db/initialize.sql
```

- h. If you are also using MWS, create and grant permissions for the MWS user to the database.

```
psql -d moab_insight
CREATE USER mws WITH PASSWORD 'changeme!';
GRANT SELECT ON ALL TABLES IN SCHEMA public TO mws;
ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO mws;
\q
```

9. On the Moab server, configure Moab's connection to Insight.

- a. In `/opt/moab/etc/moab.cfg`, configure the `INSIGHTENDPOINT` parameter so that Moab can connect to the Insight database.

```
INSIGHTENDPOINT <hostname>[:<port>]
```

`<hostname>` is the server where Insight is located. `<port>` is optional and `<hostname>` is required.

- b. Restart Moab in order for the new `INSIGHTENDPOINT` parameter to take affect.

```
service moab restart
```

- c. Verify that Moab is properly configured to connect to Insight.

```
mdiag -S | grep Insight
```

You should see something similar to the following:

```
[root]# mdiag -S | grep Insight
ZeroMQ Insight connection is bound on port 5574 (reliability port 5575) on host
* using Insight endpoint <the insight hostname displays here>:5568
encryption is on)
ZeroMQ Insight reliable message delivery is using store file(s) up to 1024 MB in
/opt/moab/spool/insight_store/
```

10. configure the message queue secret key so that Moab can connect to the Insight database.

```
INSIGHTENDPOINT <hostname>[:<port>]
```

*<hostname>* is the server where Insight is located. *<port>* is optional and *<hostname>* is required.

11. On the Insight server, configure the `moab.host` and `messageQueue.secretKey` parameters in the Insight configuration file `/opt/insight/etc/config.groovy`.

```
moab.host = "<moab server>"
messageQueue.secretKey = "<secret key>"
```

The *<secret key>* must match the secret key configured in `moab-private.cfg` on the Moab server for the **MESSAGEQUEUESECRETKEY** configuration parameter. See [Appendix A: Moab Parameters on page 923](#) in Moab Workload Manager for parameter information.

12. On the Insight server, verify that Insight and PostgreSQL run on startup.

```
[root]# chkconfig insight on
[root]# chkconfig postgresql-9.3 on
```

13. On the Insight server, start Insight.

```
[root]# service insight start
```

#### Related topics

- About Insight
- [Appendix A: Moab Parameters on page 923](#)

# Installing Viewpoint

## Requirements

### Hardware requirements:

- Dual or Quad core Intel/AMD x86-64 processor
- At least 4 GB of RAM
- At least 20 GB disk space

### Supported operating systems:

Viewpoint has been tested on the following variants of Linux:

- CentOS (6.5)
- RHEL (6.5)
- Scientific Linux (6.5)

### Supported browsers:

To run Moab Viewpoint, you must use one of these web browsers:

- Mozilla Firefox 25+
- Internet Explorer 10+
- Chrome 35+

**i** The new user interface was built on Django, a forward-thinking web framework, which relies heavily on Python; thus, HPC administrators should install Viewpoint only on systems with standard system level Python installed. The system you select for Viewpoint should not have any modifications made to its default Python installation.

## To install Viewpoint

**i** Only an RPM-based installation is supported for installing Moab Viewpoint. If you are installing Moab Viewpoint on the same server as you have installed other Moab HPC Suite RPMs, then steps 1 through 4 may be skipped. If you are installing Moab Viewpoint on a server where you used manual installation instructions to install Moab HPC Suite components, then steps 1 through 4 are necessary to install Moab Viewpoint.

1. On the Viewpoint server, download the latest 8.0.0 RPM suite tarball (`moab-hpc-basic-suite-<version>-<timestamp>-<OS>.tar.gz`, for example) from the [Adaptive Computing website](#).
2. On the Viewpoint server, untar the downloaded package.

```
[root]# tar xzf moab-hpc-basic-suite-<version>-<timestamp>-<OS>.tar.gz
```

3. On the Viewpoint server, change directories into the untarred directory.

- On the Viewpoint server, install the suite repositories. The `-y` option installs with the default settings for the RPM suite.

**i** For a description of the options of the repository installer script, run:

```
[root]# ./install-rpm-repos.sh -h
```

```
[root]# ./install-rpm-repos.sh [<repository-directory>] [-y]
```

**i** If the installation returns the following warning line:

```
Warning: RPMDB altered outside of yum.
```

This is normal and can safely be ignored.

The script installs the `createrepo` package and its dependencies. You must answer "y" to all the questions in order for the RPM install of the suite to work. Additionally, the script installs the EPEL and 10gen repositories.

- On the Viewpoint server, install the Viewpoint RPM. This will also install Apache Server and several Python packages (see the [Requirements on page 69](#) for details):

```
[root]# yum install viewpoint
```

- (Optional) The Viewpoint RPM installs an `/etc/httpd/conf.d/viewpoint.conf` file for Apache. Administrators can configure virtual hosts as required. Virtual host configurations should be made within this file. See <http://httpd.apache.org/docs/2.2/vhosts/> for more details.
  - Start the Apache service.
- ```
[root]# chkconfig httpd on
[root]# service httpd start
```
- Navigate your browser to the Viewpoint Login page `http://<viewpoint_server>/`.
  - Log in with the default credentials. The default user name is `moab-admin` and the default password is `changeme!`.
  - Proceed to About Configuration.

#### Related topics

- [Requirements on page 11](#)
- About Configuration
- Logging in/signing out

## Additional configuration

### Configuring SSL in Tomcat

To configure SSL in Tomcat, please refer to the Apache Tomcat [documentation](#).

Related topics

### Setting up OpenLDAP on CentOS 6

These instructions are intended to help first-time LDAP administrators get up and running. The following procedures contain instructions for getting started using OpenLDAP on a CentOS 6 system. For more complete information on how to set up OpenLDAP see the [OpenLDAP documentation](#).

- [Installing and configuring OpenLDAP on Centos 6 on page 71](#)
- [Adding an organizational unit \(OU\) on page 75](#)
- [Adding a user on page 76](#)
- [Adding a group on page 76](#)
- [Adding a user to a group on page 77](#)



Adaptive Computing is not responsible for creating, maintaining, or supporting customer LDAP or Active Directory configurations.

### Installing and configuring OpenLDAP on Centos 6

First, you will need to install OpenLDAP. These instructions explain how you can do this on a CentOS 6 system.

#### To install and configure OpenLDAP on Centos 6

1. Run the following command:

```
[root]# yum -y install openldap openldap-clients openldap-servers
```

2. Generate a password hash to be used as the admin password. This password hash will be used when you create the root user for your LDAP installation. For example:

```
[root]# slappasswd
New password : p@ssw0rd
Re-enter new password : p@ssw0rd
{SSHA}5lPFVw19zeh7LT53hQH69znzj8TuBrLv
```

3. Add the root user and the root user's password hash to the OpenLDAP configuration in the `olcDatabase={2}bdb.ldif` file. The root user will have permissions to add other users, groups,

organizational units, etc. Do the following:

- a. Run this command:

```
[root]# cd /etc/openldap/slapd.d/cn=config
[root]# vi olcDatabase=\{2\}bdb.ldif
```

- b. If the **olcRootPW** attribute does not already exist, create it. Then set the value to be the hash you created from `slappasswd`. For example:

```
olcRootPW: {SSHA}51PFVw19zeh7LT53hQH69znzj8TuBrLv
...
```

4. While editing this file, change the distinguished name (DN) of the **olcSuffix** to something appropriate. The suffix typically corresponds to your DNS domain name, and it will be appended to the DN of every other LDAP entry in your LDAP tree.

For example, let's say your company is called Acme Corporation, and that your domain name is "acme.com." You might make the following changes to the `olcDatabase={2}bdb.ldif` file:

```
olcSuffix: dc=acme,dc=com
...
olcRootDN: cn=Manager,dc=acme,dc=com
...
olcRootPW: {SSHA}51PFVw19zeh7LT53hQH69znzj8TuBrLv
...
```



Throughout the following examples in this topic, you will see `dc=acme,dc=com`. "acme" is only used as an example to illustrate what you would use as your own domain controller if your domain name was "acme.com." You should replace any references to "acme" with your own organization's domain name.



Do not set the cn of your root user to "root" (`cn=root,dc=acme,dc=com`), or OpenLDAP will have problems.

5. Modify the DN of the root user in the `olcDatabase={1}monitor.ldif` file to match the **olcRootDN** line in the `olcDatabase={2}bdb.ldif` file. Do the following:

- a. Run this command to edit the `olcDatabase={2}bdb.ldif` file:

```
[root]# vi olcDatabase=\{1\}monitor.ldif
```

- b. Modify the **olcAccess** line so that the **dn.base** matches the **olcRootDN** from the `olcDatabase={2}bdb.ldif` file. (In this example, **dn.base** should be "cn=Manager,dc=acme,dc=com".)

```
olcAccess: {0}to * by
dn.base="gidNumber=0+uidNumber=0,cn=peercred,cn=external,cn=auth" read by
dn.base="cn=Manager,dc=acme,dc=com" read by * none
```

- c. Now the root user for your LDAP is `cn=Manager,dc=acme,dc=com`. The root user's password is the password that you entered using `slappasswd` (see step 2), which, in this example, is **p@ssw0rd**
6. Hide the password hashes from users who should not have permission to view them.

**i** A full discussion on configuring access control in OpenLDAP is beyond the scope of this tutorial. For help, see [the OpenLDAP Access Control documentation](#).

- a. Run this command to edit the `olcDatabase=\{2\}bdb.ldif` file:

```
[root]# vi olcDatabase=\{2\}bdb.ldif
```

- b. Add the following two lines to the end of the file to restrict users from viewing other users' password hashes.

```
olcAccess: {0}to attrs=userPassword by self write by
dn.base="cn=Manager,dc=acme,dc=com" write by anonymous auth by * none
olcAccess: {1}to * by dn.base="cn=Manager,dc=acme,dc=com" write by self write by
* read
```

*These lines allow a user to read and write his or her own password. It also allows a manager to read and write anyone's password. Anyone, including anonymous users, is allowed to view non-password attributes of other users.*

7. Make sure that OpenLDAP is configured to start when the machine starts up, and start the OpenLDAP service.

```
[root]# chkconfig slapd on
[root]# service slapd start
```

8. Now, you must manually create the "dc=acme,dc=com" LDAP entry in your LDAP tree.

An LDAP directory is analogous to a tree. Nodes in this tree are called LDAP "entries" and may represent users, groups, organizational units, domain controllers, or other objects. The attributes in each entry are determined by the LDAP schema. In this tutorial we will build entries based on the `InetOrgPerson` schema (which ships with OpenLDAP by default).

In order to build our LDAP tree we must first create the root entry. Root entries are usually a special type of entry called a domain controller (DC). Because we are assuming that the organization is called Acme Corporation, and that the domain is "acme.com," we will create a domain controller LDAP entry called `dc=acme,dc=com`. Again, you will need to replace "acme" with your organization's domain name. Also note that `dc=acme,dc=com` is what is called an LDAP distinguished name (DN). An LDAP distinguished name uniquely identifies an LDAP entry.

Do the following:

- a. Create a file called `acme.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi acme.ldif
```

- b. Add the following lines in `acme.ldif`:

```
dn: dc=acme,dc=com
objectClass: dcObject
objectClass: organization
dc: acme
o : acme
```

- c. Now add the contents of this file to LDAP. Run this command:

```
[root]# ldapadd -f acme.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

- d. Verify that your entry was added correctly.

```
[root]# ldapsearch -x -LLL -b dc=acme,dc=com
dn: dc=acme,dc=com
objectClass: dcObject
objectClass: organization
dc: acme
o: acme
```

9. Run the following:

```
[root]# sudo iptables -L
[root]# sudo service iptables save
```

10. By default, the CentOS 6 firewall will block external requests to OpenLDAP. In order to allow MWS to access LDAP, you will have to configure your firewall to allow connections on port 389. (Port 389 is the default LDAP port.)

Configuring your firewall is beyond the scope of this tutorial; however, it may be helpful to know that the default firewall on CentOS is a service called `iptables`. (For more information, see the documentation on [iptables](#).) In the most basic case, you may be able to add a rule to your firewall that accepts connections to port 389 by doing the following:

- a. Edit your `iptables` file:

```
[root]# vi /etc/sysconfig/iptables
```

- b. Add the following line *after* all the **ACCEPT** lines but *before* any of the **REJECT** lines in your `iptables` file:

```
# ... lines with ACCEPT should be above
-A INPUT -p tcp --dport 389 -j ACCEPT
# .. lines with REJECT should be below
```

For example, here is a sample `iptables` file with this line added:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -p tcp --dport 389 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited

COMMIT
```

c. Now reload iptables.

```
[root]# service iptables reload
```



Although providing instructions is beyond the scope of this tutorial, it is also highly recommended that you set up OpenLDAP to use SSL or TLS security to prevent passwords and other sensitive data from being sent in plain text. For information on how to do this, see the [OpenLDAP TLS documentation](#).

Now that you have installed and set up Open LDAP, you are ready to add organizational units (see [Adding an organizational unit \(OU\) on page 75](#)).

## Adding an organizational unit (OU)

These instructions will describe how to populate the LDAP tree with organizational units (OUs), groups, and users, all of which are different types of LDAP entries. The examples that follow also presume an InetOrgPerson schema, because the InetOrgPerson schema is delivered with OpenLDAP by default.

### To add an organizational unit (OU) entry to the LDAP tree

In this example, we are going to add an OU called "Users."

1. Create a temporary file called `users.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi users.ldif
```

2. Add these lines to `users.ldif`:

```
dn: ou=Users,dc=acme,dc=com
objectClass: organizationalUnit
ou: Users
```

3. Add the contents of `users.ldif` file to LDAP.

```
[root]# ldapadd -f users.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a user

### To add a user to LDAP

In this example, we will add a user named "Bob Jones" to LDAP inside the "Users" OU.

1. Create a temporary file called `bob.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi bob.ldif
```

2. Add these lines to `bob.ldif`:

```
dn: cn=Bob Jones,ou=Users,dc=acme,dc=com
cn: Bob Jones
sn: Jones
objectClass: inetOrgPerson
userPassword: p@ssw0rd
uid: bjones
```

3. Add the contents of `bob.ldif` file to LDAP.

```
[root]# ldapadd -f bob.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a group

### To add a group to LDAP

In this example, we will add a group called "Engineering" to LDAP inside the "Users" OU.

1. Create a temporary file called `engineering.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi engineering.ldif
```

2. Add these lines to `engineering.ldif`:

```
dn: cn=Engineering,ou=Users,dc=acme,dc=com
cn: Engineering
objectClass: groupOfNames
member: cn=Bob Jones,ou=Users,dc=acme,dc=com
```

3. Add the contents of `engineering.ldif` file to LDAP.

```
[root]# ldapadd -f engineering.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

## Adding a user to a group

### To add a user to an LDAP group

In this example, we will add an LDAP member named "Al Smith" to the "Engineering" LDAP group. This example assumes that user, Al Smith, has already been added to LDAP.

**i** Before you add a user to an LDAP group, the user must first be added to LDAP. For more information, see [Adding a user on page 76](#).

1. Create a temporary file called `addUserToGroup.ldif`. (You can delete this file once its content has been added to LDAP, so in this example, we will create it in the `/tmp` folder.)

```
[root]# cd /tmp
[root]# vi addUserToGroup.ldif
```

2. Add these lines to `addUserToGroup.ldif`:

```
dn: cn=Engineering,ou=Users,dc=acme,dc=com
changetype: modify
add: member
member: cn=Al Smith,ou=Users,dc=acme,dc=com
```

3. Now add the contents of `addUserToGroup.ldif` file to LDAP.

```
[root]# ldapadd -f addUserToGroup.ldif -D cn=Manager,dc=acme,dc=com -w p@ssw0rd
```

Related topics

## Trusting servers in Java

### Prerequisites

Some of these instructions refer to `JAVA_HOME`, which must point to the same directory that Tomcat uses. To set `JAVA_HOME`, do this:

```
[root]# source /etc/tomcat6/tomcat6.conf
```

Your system administrator might have defined Tomcat's `JAVA_HOME` in a different file.

### Retrieve the server's X.509 public certificate

To retrieve the server's certificate, use the following command:

```
[root]# $JAVA_HOME/bin/keytool -printcert -rfc -sslserver <servername>:<port> >
/tmp/public.cert.pem
```

Replace `<servername>` with the server's host name and `<port>` with the secure port number. The default port for https is 443. The default port for ldaps is 636. If successful, `/tmp/public.cert.pem` contains the server's public certificate. Otherwise, `/tmp/public.cert.pem` contains an error message. This

message is typical: `keytool error: java.lang.Exception: No certificate from the SSL server`. This message suggests that the server name or port is incorrect. Consult your IT department to determine the correct server name and port.

## Add the server's certificate to Java's keystore

Java stores trusted certificates in a database known as the keystore. Because each new version of Java has its own keystore, you need to add the server certificate to the Java keystore (using the steps below) every time you install a new version of Java.

Java's keystore is located at `$JAVA_HOME/lib/security/cacerts`. If Tomcat's `JAVA_HOME` points to a JDK, then the keystore is located at `$JAVA_HOME/jre/lib/security/cacerts`. To add the server certificate to the keystore, run the following command:

```
[root]# $JAVA_HOME/bin/keytool -import -trustcacerts -file /tmp/public.cert.pem -alias
<servername> -keystore $JAVA_HOME/lib/security/cacerts
```

You will be prompted for the keystore password, which is "changeit" by default.

 Your system administrator might have changed this password.

After you've entered the keystore password, you'll see the description of the server's certificate. At the end of the description it prompts you to trust the certificate.

```
Trust this certificate? [no]:
```

Type `yes` and press **Enter** to add the certificate to the keystore.

# Troubleshooting

This page details some common problems and general solutions. It contains these sections:

- [General issues on page 78](#)
- [Moab Web Services issues on page 82](#)

## General issues

- [Moab error: "cannot determine local hostname" on page 79](#)
- [Moab error: "Moab will now exit due to license file not found" on page 79](#)
- [Other Moab issues on page 79](#)
- [Where do I change my passwords? on page 80](#)

### Moab error: "cannot determine local hostname"

```
# service moab start
Starting moab: ERROR:      cannot determine local hostname - node is misconfigured
                        [FAILED]
```

If you encounter this error when starting Moab, check the `/opt/moab/etc/moab.cfg` file to make sure a valid host is configured. For example:

```
...
SCHEDCFG[Moab]                SERVER=<moab-hostname>:42559
...
```

Also check `/etc/hosts` to be sure the host name resolves, at least with localhost:

```
...
127.0.0.1    <moab-hostname> localhost localhost.localdomain localhost4
localhost4.localhostdomain4
...
```

### Moab error: "Moab will now exit due to license file not found"

```
# service moab start
Starting moab: Moab will now exit due to license file not found
Please contact Adaptive Computing (sales@adaptivecomputing.com) to get a license
for your system
                        [FAILED]
```

If you encounter this error when starting Moab, make sure your Moab license file is named **moab.lic** and is located in the `/opt/moab/etc/` directory.

Also make sure the license is not expired. The expiration date is listed in the license file. For example:

```
# cat /opt/moab/etc/moab.lic
...
# Expires after Tue Dec 31 10:43:46 2013
...
```

### Other Moab issues

Please see "Troubleshooting and System Maintenance" in the *Moab Workload Manager Administrator Guide*.

[Where do I change my passwords?](#)

**MWS super user username and password**

The default username and password for MWS are **moab-admin** and **changeme!** (respectively).

To change the username and/or the password for the MWS super user:

1. Stop the tomcat6 and moab services.

```
[root]# service moab stop
[root]# service tomcat6 stop
```

2. Change the respective values in the following files:

- /opt/mws/etc/mws-config.groovy:

```
auth.defaultUser.username = "moab-admin"
auth.defaultUser.password = "changeme!"
```

- /opt/moab/etc/moab-private.cfg:

```
CLIENTCFG[RM:mws] USERNAME=moab-admin PASSWORD=changeme!
```

- /opt/moab/etc/cloud.cfg:

```
CONFIG[default] MWS_USERNAME=moab-admin
CONFIG[default] MWS_PASSWORD=changeme!
```

3. Start the tomcat6 service.

```
[root]# service tomcat6 start
```

4. Start the moab service.

```
[root]# service moab start
```

**MongoDB passwords**

To change the passwords for MongoDB:

1. Stop the tomcat6 and moab services.

```
[root]# service moab stop
[root]# service tomcat6 stop
```

2. Change the passwords for the MongoDB accounts (i.e., **moab\_user** and/or **mws\_user**). For instructions, see the [MongoDB documentation](#).

3. Edit the password values in the following files:

- /opt/moab/etc/moab-private.cfg:

```
MONGouser moab_user
MONGOPASSWORD secret2
```

- /opt/mws/etc/mws-config.groovy:

```
// MongoDB configuration.  
grails.mongo.username = "mws_user"  
grails.mongo.password = "secret3"
```

4. Start the tomcat6 service.

```
[root]# service tomcat6 start
```

5. Start the moab service.

```
[root]# service moab start
```

## Moab Web Services issues

If something goes wrong with MWS, look in the following files:

- The MWS log file. By default this is `/opt/mws/log/mws.log`.
- The Tomcat `catalina.out` file, usually in `/var/log/tomcat6` or `$CATALINA_HOME/logs`.

**i** If you remove the `log4j` configuration from `mws-config.groovy`, MWS writes its log files to `java.io.tmpdir`. For Tomcat, `java.io.tmpdir` is generally set to `$CATALINA_BASE/temp` or `CATALINA_TMPDIR`.

Here is a list of some errors and their fixes:

- [MongoDB: Errors during MWS startup on page 83](#)
- [MongoDB: Out of semaphores to get db connection on page 85](#)
- [MongoDB: Connection wait timeout after 120000 ms on page 85](#)
- [java.lang.OutOfMemoryError: Java heap space on page 85](#)
- [java.lang.OutOfMemoryError: PermGen space on page 86](#)
- [SEVERE: Context \[/mws\] startup failed due to previous errors on page 86](#)
- [Moab Reached Maximum Number of Concurrent Client Connections on page 86](#)

[MongoDB: Errors during MWS startup](#)

If the application fails to start and gives error messages such as these:

```
Error creating bean with name 'mongoDatastore'
can't say something; nested exception is com.mongodb.MongoException
```

```
ERROR    grails.app.services.com.ace.mws.ErrorService    0
Error encountered while attempting to authenticate account or query database;
the MongoDB server is not available. Please verify connection to server
'/127.0.0.1:27017' and that MongoDB is running.
```

MongoDB is most likely not running, or the MongoDB host and port are misconfigured.

In this case, there are a few things to verify:

- (Not relevant if MongoDB is installed on a separate server) **Is MongoDB installed?**

Run the following commands to assess whether MongoDB is installed on the current server.

```
$ mongo
-bash: mongo: command not found
```

To remedy, install MongoDB, start the `mongod` service and then restart the `tomcat6` service. See [Preparing for installation on page 17](#) or [Installing Moab HPC Suite - Basic Edition on page 54](#) for more information on how to install and configure MongoDB in the manual installation guide or the RPM-based installation guide, respectively.

- (Only relevant if MongoDB is installed on a separate server) **Is MWS configured to connect to the remote MongoDB server?**

Run the following commands to assess whether MongoDB is installed on the current server.

```
[root]# cat /opt/mws/etc/mws-config.groovy | grep 'grails.mongo'
// grails.mongo.username = "mws_user"
// grails.mongo.password = "<ENTER-KEY-HERE>"
// grails.mongo.host = "127.0.0.1"
// grails.mongo.port = 27017
```

Make sure that the `grails.mongo.*` options are configured in `/opt/mws/etc/mws-config.groovy` for the remote MongoDB server and then restart the `tomcat6` service.

```
[root]# service tomcat6 restart
```

- **Is MWS configured to authenticate with MongoDB, and is MongoDB configured to enforce authentication?**

Run the following commands to assess the relevant MWS and MongoDB configurations.

```
[root]# cat /opt/mws/etc/mws-config.groovy | grep 'grails.mongo'
// grails.mongo.username = "mws_user"
// grails.mongo.password = "<ENTER-KEY-HERE>"

[root]# cat /etc/mongod.conf | grep 'auth'
#noauth = true
auth = true
```

The configuration above is problematic because the `grails.mongo` credentials are commented out in the `/opt/mws/etc/mws-config.groovy` file while MongoDB is

configured to enforce authentication ("auth = true"). Similar connection issues will exist if the `grails.mongo` parameters do not match the credentials configured for the "mws\_user" on both the `mws` and `moab` databases in MongoDB.

(For upgrade scenarios only) If the application fails to start and gives the following message in `/opt/mws/etc/log/mws.log`:

```
java.lang.Exception: The db-migrate.js script has not yet been run. Please see the
upgrade section of the installation guide for instructions.
```

Then the `db-migrate.js` script must be run to update the schema of the `mws` database in MongoDB.

### MongoDB: Out of semaphores to get db connection

To resolve this error, adjust the values of `connectionsPerHost` or `threadsAllowedToBlockForConnectionMultiplier` by adding them to `mws-config.groovy`. Example:

```
grails.mongo.options.connectionsPerHost = 60
grails.mongo.options.threadsAllowedToBlockForConnectionMultiplier = 10
```

For more information on these options, refer to these documents:

- "Configuration" (in the Moab Web Services *Reference Guide*), which briefly discusses a few MongoDB driver options.
- The [MongoOptions](#) documentation, which contains full details on all MongoDB driver options.

**i** You must restart Tomcat after adding, removing, or changing `grails.mongo.options` parameters.

As shipped, `mws-config.groovy` does not contain any `grails.mongo.options` parameters. To adjust their values, you need to add them to `mws-config.groovy`.

The default value of `connectionsPerHost` is normally 10, but MWS sets it internally to 50.

The default value of `threadsAllowedToBlockForConnectionMultiplier` is 5.

Any of the options listed in `MongoOptions` can be specified in `mws-config.groovy`. Just use the prefix `grails.mongo.options` as shown above.

### MongoDB: Connection wait timeout after 120000 ms

See [MongoDB: Out of semaphores to get db connection](#) above.

### java.lang.OutOfMemoryError: Java heap space

Increase the size of the heap using JVM options `-Xms` and `-Xmx`. Here are the suggested values:

```
CATALINA_OPTS="-DMWS_HOME=/opt/mws -Xms256m-Xmx3g -XX:MaxPermSize=384m"
```

- `-Xms`: Set initial Java heap size.
- `-Xmx`: Set maximum Java heap size.

### java.lang.OutOfMemoryError: PermGen space

Increase the size of the permanent generation using JVM option **-XX:MaxPermSize**. Here are the suggested values:

```
CATALINA_OPTS="-DMWS_HOME=/opt/mws -Xms256m -Xmx3g -XX:MaxPermSize=384m"
```

### SEVERE: Context [/mws] startup failed due to previous errors

If `catalina.out` contains this error, look in `/opt/mws/log/mws.log` and `/opt/mws/log/stacktrace.log` for more details on the error.

Also ensure that the `/opt/mws/etc/mws-config.groovy` file can be read by the Tomcat user. The permissions should appear as follows:

```
$ ls -al /opt/mws/etc/mws-config.groovy
-r----- 1 tomcat tomcat 4056 Dec  4 12:07 mws-config.groovy
```

### Moab Reached Maximum Number of Concurrent Client Connections

When this error message is encountered, simply add a new line to the `moab.cfg` file:

```
CLIENTMAXCONNECTIONS 256
```

This will change the Moab configuration when Moab is restarted. Run the following command to immediately use the new setting:

```
[root]# changeparam CLIENTMAXCONNECTIONS 256
```

 The number **256** above may be substituted for the desired maximum number of Moab client connections.

## Component documentation

The individual components of the suite have more options and allow for more configuration than can be contained in this guide. Refer to the individual component guides for more information.

### TORQUE

- TORQUE 5.0.0 Administrator Guide: [HTML](#)  - [PDF](#) 

### Moab Workload Manager

- Moab Workload Manager 8.0.0 Administrator Guide: [HTML](#) 

### Moab Web Services

- Moab Web Services 8.0.0 Reference Guide: [HTML](#) 

### Related topics

- [Preparing for installation on page 17](#)
- [Installing Moab HPC Suite - Basic Edition on page 54](#)

- [Welcome on page xxiii](#)



# Moab Workload Manager

## Moab Workload Manager overview

Moab Workload Manager is a highly advanced scheduling and management system designed for clusters, grids, and on-demand/utility computing systems. At a high level, Moab applies site policies and extensive optimizations to orchestrate jobs, services, and other workload across the ideal combination of network, compute, and storage resources. Moab enables true adaptive computing allowing compute resources to be customized to changing needs and failed systems to be automatically fixed or replaced. Moab increases system resource availability, offers extensive cluster diagnostics, delivers powerful QoS/SLA features, and provides rich visualization of cluster performance through advanced statistics, reports, and charts.

Moab works with virtually all major resource management and resource monitoring tools. From hardware monitoring systems like IPMI to provisioning systems and storage managers, Moab takes advantage of domain expertise to allow these systems to do what they do best, importing their state information and providing them with the information necessary to better do their job. Moab uses its global information to coordinate the activities of both resources and services, which optimizes overall performance in-line with high-level mission objectives.

Related topics

- 

## Philosophy

The scheduler's purpose is to optimally use resources in a convenient and manageable way. System users want to specify resources, obtain quick turnaround on their jobs, and have reliable resource allocation. On the other hand, administrators want to understand both the workload and the resources available. This includes current state, problems, and statistics—information about what is happening that is transparent to the end-user. Administrators need an extensive set of options to enable management enforced policies and tune the system to obtain desired statistics.

There are other systems that provide batch management; however, Moab is unique in many respects. Moab matches jobs to nodes, dynamically reprovisions nodes to satisfy workload, and dynamically modifies workload to better take advantage of available nodes. Moab allows sites to fully visualize cluster and user behavior. It can integrate and orchestrate resource monitors, databases, identity managers, license managers, networks, and storage systems, thus providing a cohesive view of the cluster—a cluster that fully acts and responds according to site mission objectives.

Moab can dynamically adjust security to meet specific job needs. Moab can create real and virtual clusters on demand and from scratch that are custom-tailored to a specific request. Moab can integrate visualization services, web farms and application servers; it can create powerful grids of disparate clusters. Moab maintains complete accounting and auditing records, exporting this data to information services on command, and even providing professional billing statements to cover all used resources and services.

Moab provides user- and application-centric web portals and powerful graphical tools for monitoring and controlling every conceivable aspect of a cluster's objectives, performance, workload, and usage. Moab is unique in its ability to deliver a powerful user-centric cluster with little effort. Its design is focused on ROI, better use of resources, increased user effectiveness, and reduced staffing requirements.

This chapter contains these sections:

- [Value of a Batch System on page 90](#)
- [Philosophy and Goals on page 91](#)
- [Workload on page 92](#)

## Value of a Batch System

Batch systems provide centralized access to distributed resources through mechanisms for submitting, launching, and tracking jobs on a shared resource. This greatly simplifies use of the cluster's distributed resources, allowing users a *single system image* in terms of managing jobs and aggregate compute resources available. Batch systems should do much more than just provide a global view of the cluster, though. Using compute resources in a fair and effective manner is complex, so a scheduler is necessary to determine when, where, and how to run jobs to optimize the cluster. Scheduling decisions can be categorized as follows:

- [Traffic Control](#)
- [Mission Policies](#)
- [Optimizations](#)

### Traffic Control

A scheduler must prevent jobs from interfering. If jobs contend for resources, cluster performance decreases, job execution is delayed, and jobs may fail. Thus, the scheduler tracks resources and dedicates requested resources to a particular job, which prevents use of such resources by other jobs.

### Mission Policies

Clusters and other HPC platforms typically have specific purposes; to fulfill these purposes, or mission goals, there are usually rules about system use pertaining to who or what is allowed to use the system. To be effective, a scheduler must provide a suite of policies allowing a site to *map* site mission policies into scheduling behavior.

## Optimizations

The compute power of a cluster is a limited resource; over time, demand inevitably exceeds supply. Intelligent scheduling decisions facilitate higher job volume and faster job completion. Though subject to the constraints of the traffic control and mission policies, the scheduler must use whatever freedom is available to maximize cluster performance.

## Philosophy and Goals

Managers want high system utilization and the ability to deliver various qualities of service to various users and groups. They need to understand how available resources are delivered to users over time. They also need administrators to tune *cycle delivery* to satisfy the current site mission objectives.

Determining a scheduler's success is contingent upon establishing metrics and a means to measure them. The value of statistics is best understood if optimal statistical values are known for a given environment, including workload, resources, and policies. That is, if an administrator could determine that a site's typical workload obtained an average queue time of 3.0 hours on a particular system, that would be a useful *statistic*; however, if an administrator knew that through proper tuning the system could deliver an average queue time of 1.2 hours with minimal negative side effects, that would be valuable *knowledge*.

Moab development relies on extensive feedback from users, administrators, and managers. At its core, it is a tool designed to *manage* resources and provide meaningful information about what is actually happening on the system.

## Management Goals

A manager must ensure that a cluster fulfills the purpose for which it was purchased, so a manager must deliver cycles to those projects that are most critical to the success of the funding organizations.

Management tasks to fulfill this role may include the following:

- Define cluster mission objectives and performance criteria
- Evaluate current and historical cluster performance
- Instantly graph delivered service

## Administration Goals

An administrator must ensure that a cluster is effectively functioning within the bounds of the established mission goals. Administrators translate goals into cluster policies, identify and correct cluster failures, and train users in best practices. Given these objectives, an administrator may be tasked with each of the following:

- Maximize utilization and cluster responsiveness
- Tune fairness policies and workload distribution
- Automate time-consuming tasks
- Troubleshoot job and resource failures

- Instruct users of available policies and in their use regarding the cluster
- Integrate new hardware and cluster services into the batch system

## End-user Goals

End-users are responsible for learning about the resources available, the requirements of their workload, and the policies to which they are subject. Using this understanding and the available tools, they find ways to obtain the best possible responsiveness for their own jobs. A typical end-user may have the following tasks:

- Manage current workload
- Identify available resources
- Minimize workload response time
- Track historical usage
- Identify effectiveness of prior submissions

## Workload

Moab can manage a broad spectrum of compute workload types, and it can optimize all four workload types within the same cluster simultaneously, delivering on the objectives most important to each workload type. The workload types include the following:

- [Batch Workload](#)
- [Interactive Workload](#)
- [Calendar Workload](#)
- [Service Workload](#)

### Batch Workload

Batch workload is characterized by a *job* command file that typically describes all critical aspects of the needed compute resources and execution environment. With a batch job, the job is submitted to a job queue, and is run somewhere on the cluster as resources become available. In most cases, the submitter will submit multiple batch jobs with no execution time constraints and will process the job results as they become available.

Moab can enforce rich policies defining how, when, and where batch jobs run to deliver compute resources to the most important workload and provide general SLA guarantees while maximizing system utilization and minimizing average response time.

### Interactive Workload

Interactive workload differs from batch in that requestors are interested in immediate response and are generally waiting for the interactive request to be executed before going on to other activities. In many cases, interactive submitters will continue to be *attached* to the interactive job, routing keystrokes and

other input into the job and seeing both output and error information in real-time. While interactive workload may be submitted within a job file, commonly, it is routed into the cluster via a web or other graphical terminal and the end-user may never even be aware of the underlying use of the batch system.

For managing interactive jobs, the focus is usually on setting aside resources to guarantee immediate execution or at least a minimal wait time for interactive jobs. Targeted service levels require management when mixing batch and interactive jobs. Interactive and other jobs types can be dynamically steered in terms of what they are executing as well as in terms of the quantity of resources required by the application. Moab can apply dynamic or malleable job facilities to dynamically grow and shrink jobs as needed to meet these changing constraints.

## Calendar Workload

Calendar workload must be executed at a particular time and possibly in a regular periodic manner. For such jobs, time constraints range from flexible to rigid. For example, some calendar jobs may need to complete by a certain time, while others must run exactly at a given time each day or each week.

Moab can schedule the future and can thus guarantee resource availability at needed times to allow calendar jobs to run as required. Furthermore, Moab provisioning features can locate or temporarily create the needed compute environment to properly execute the target applications.

## Service Workload

Moab can schedule and manage both individual applications and long-running or persistent services. Service workload processes externally-generated transaction requests while Moab provides the distributed service with needed resources to meet target backlog or response targets to the service. Examples of service workload include parallel databases, web farms, and visualization services. Moab can apply cluster, [grid](#), or dynamically-generated on-demand resources to the service.

When handling service workload, Moab observes the application in a highly abstract manner. Using the [JOB\\_CFG](#) parameter, aspects of the service jobs can be discovered or configured with attributes describing them as resource consumers possessing response time, backlog, state metrics, and associated QoS targets. In addition, each application can specify the type of compute resource required (OS, arch, memory, disk, network adapter, data store, and so forth) as well as the support environment (network, storage, external services, and so forth).

If the QoS response time/backlog targets of the application are not being satisfied by the current resource allocation, Moab evaluates the needs of this application against all other site mission objectives and workload needs and determines what it must do to locate or create (that is, provision, customize, secure) the needed resources. With the application resource requirement specification, a site may also indicate proximity/locality constraints, partition policies, ramp-up/ramp-down rules, and so forth.

Once Moab identifies and creates appropriate resources, it hands these resources to the application via a site customized URL. This URL can be responsible for whatever application-specific hand-shaking must be done to launch and initialize the needed components of the distributed application upon the new resources. Moab engages in the hand-off by providing needed context and resource information and by launching the URL at the appropriate time.

## Related topics

- [Malleable Jobs](#)
- [QOS/SLA Enforcement](#)

## Scheduler Basics

- [Initial Moab Configuration](#) on page 94
- [Layout of Scheduler Components](#) on page 96
- [Scheduling Environment](#) on page 98
  - [Scheduling Dictionary](#) on page 104
- [Scheduling Iterations and Job Flow](#) on page 111
- [Configuring the Scheduler](#) on page 114
- [Credential Overview](#) on page 117
  - [Job Attributes/Flags Overview](#) on page 144

## Initial Moab Configuration

### Configuring an RPM-based install of Moab

When Moab is installed via an RPM source, such as with the Moab HPC Suite or Moab Cloud Suite, the `moab.cfg` file contains only one directive - an `#IMPORT` line that imports all the configuration files in `/opt/moab/etc`. The usual configuration settings that are normally contained in `moab.cfg` have been moved to `moab-server.cfg`. Moab still reads the `moab.cfg` file and, due to the `#INCLUDE` directive, reads in all the other configuration files as well.

To configure Moab in the case of an RPM install, you can modify the `moab.cfg` file, the `moab-server.cfg` file, or any of the configuration files that are read in by `moab.cfg` such as the accounting manager configuration file (`am.cfg`) or the resource manager configuration file (`rm.cfg`).

The RPMs allow for a client install of Moab, instead of a server install. In this instance, the `moab-server.cfg` file is replaced with a `moab-client.cfg` file. The server and client RPMs cannot be installed on the same machine.

### Basic configuration of Moab

After Moab is installed, there may be minor configuration remaining within the primary configuration file, `moab.cfg`. While the `configure` script automatically sets these parameters, sites may choose to specify additional parameters. If the values selected in `configure` are satisfactory, then this section may be safely ignored.

The parameters needed for proper initial startup include the following:

| Parameter                | Instructions                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">SCHEDCFG</a> | <p>The <b>SCHEDCFG</b> parameter specifies how the Moab server will execute and communicate with client requests. The <b>SERVER</b> attribute allows Moab client commands to locate the Moab server and is specified as a URL or in <code>&lt;HOST&gt;[:&lt;PORT&gt;]</code> format. For example:</p> <pre>SCHEDCFG[orion] SERVER=cw.psu.edu</pre> <p>Specifying the server in the Moab configuration file is optional. If nothing is specified, <code>gethostname()</code> is called. You can restart Moab and run <code>mdiag -S</code> to confirm that the correct host name is specified.</p> <p><b>i</b> The <b>SERVER</b> attribute can also be set using the environment variable <b>\$MOABSERVER</b>. Using this variable allows you to quickly change to the Moab server that client commands will connect to.</p> <pre>&gt; export MOABSERVER=cluster2:12221</pre>                                                                                                             |
| <a href="#">ADMINCFG</a> | <p>Moab provides role-based security enabled via multiple levels of admin access. Users who are to be granted full control of all Moab functions should be indicated by setting the <b>ADMINCFG[1]</b> parameter. The first user in this <b>USERS</b> attribute list is considered the <i>primary</i> administrator. It is the ID under which Moab will execute. For example, the following may be used to enable users <i>greg</i> and <i>thomas</i> as level 1 admins:</p> <pre>ADMINCFG[1] USERS=greg,thomas</pre> <p><b>i</b> Moab may only be launched by the primary administrator user ID.</p> <p><b>i</b> The primary administrator should be configured as a manager/operator/administrator in every resource manager with which Moab will interface.</p> <p><b>i</b> If the <code>msub</code> command will be used, then "root" <i>must</i> be the primary administrator.</p> <p><b>i</b> Moab's home directory and contents should be owned by the primary administrator.</p> |
| <a href="#">RMCFG</a>    | <p>For Moab to properly interact with a resource manager, the interface to this resource manager must be defined as described in the <a href="#">Resource Manager Configuration Overview</a>. Further, it is important that the primary Moab administrator also be a resource manager administrator within each of those systems. For example, to interface to a <b>TORQUE</b> resource manager, the following may be used:</p> <pre>RMCFG[torque1] TYPE=pbs</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

#### Related topics

- [Parameter Overview](#)
- `mdiag -C` command (for diagnosing current Moab configuration)

## Layout of Scheduler Components

Moab is initially unpacked into a simple one-deep directory structure. What follows demonstrates the default layout of scheduler components; some of the files (such as log and statistics files) are created while Moab runs.

- \* `$(MOABHOMEDIR)` (default is `/opt/moab` and can be modified via the `--with-homedir` parameter during `./configure`) contains the following files:

| Filename                        | Description                                                          |
|---------------------------------|----------------------------------------------------------------------|
| <b>.moab.ck</b>                 | Checkpoint file                                                      |
| <b>.moab.pid</b>                | Lock file                                                            |
| <b>moab.lic</b>                 | License file                                                         |
| <b>contrib/</b>                 | Directory containing contributed code and plug-ins                   |
| <b>docs/</b>                    | Directory for documentation                                          |
| <b>etc/</b>                     | Directory for configuration files                                    |
| <b>moab.cfg</b>                 | General configuration file                                           |
| <b>moab.dat</b>                 | Configuration file generated by <a href="#">Moab Cluster Manager</a> |
| <b>moab-private.cfg</b>         | Secure configuration file containing private information             |
| <b>lib/</b>                     | Directory for library files (primarily for <code>tools/</code> )     |
| <b><a href="#">log/</a></b>     | Directory for log files                                              |
| <b><a href="#">moab.log</a></b> | Log file                                                             |
| <b>moab.log.1</b>               | Previous log file                                                    |

| Filename                 | Description                                                                                                                                                                                                                                                                                                |
|--------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">stats/</a>   | Directory for statistics files: <ul style="list-style-type: none"> <li>◦ <code>events.&lt;date&gt;</code> - event files</li> <li>◦ <code>{DAY WEEK MONTH YEAR}.&lt;date&gt;</code> - usage profiling data</li> <li>◦ <code>FS.&lt;PARTITION&gt;.&lt;epochtime&gt;</code> - fairshare usage data</li> </ul> |
| <a href="#">samples/</a> | Directory for sample configuration files, simulation trace files, etc.                                                                                                                                                                                                                                     |

- `$ (MOABINSTDIR)` (default is `/opt/moab` and can be modified via the `--prefix` parameter during `./configure`) contains the following files:

| Filename               | Description                                                                                        |
|------------------------|----------------------------------------------------------------------------------------------------|
| <a href="#">bin/</a>   | Directory for client commands (for example, <a href="#">showq</a> , <a href="#">setres</a> , etc.) |
| <a href="#">sbin/</a>  | Directory for server daemons                                                                       |
| <a href="#">moab</a>   | Moab binary                                                                                        |
| <a href="#">tools/</a> | Directory for resource manager interfaces and local scripts                                        |

- `/etc/moab.cfg` - If the Moab home directory cannot be found at startup, this file is checked to see if it declares the Moab home directory. If a declaration exists, the system checks the declared directory to find Moab. The syntax is: `MOABHOMEDIR=<DIRECTORY>`.

If you want to run Moab from a different directory other than `/opt/moab` but did not use the `--with-homedir` parameter during `./configure`, you can set the `$MOABHOMEDIR` environment variable, declare the home directory in the `/etc/moab.cfg` file, or use the `-C` command line option when using the Moab server or client commands to specify the configuration file location.

When Moab runs, it creates a log file, `moab.log`, in the `log/` directory and creates a statistics file in the `stats/` directory with the naming convention `events.WWW_MMM_DD_YYYY` (for example, `events.Sat_Oct_10_2009`). Additionally, a checkpoint file, `.moab.ck`, and lock file, `.moab.pid`, are maintained in the Moab home directory.

## Layout of Scheduler Components with Integrated Database Enabled

If [USEDATABASE INTERNAL](#) is configured, the layout of scheduler components varies slightly. The `.moab.ck` file and usage profiling data (`stat/{DAY|WEEK|MONTH|YEAR}.<date>`) are stored in the `moab.db` database. In addition, the event information is stored in both event files: (`stat/events.<date>`) and `moab.db`.

## Related topics

- [Commands Overview](#)
- Installation

# Scheduling Environment

Moab functions by manipulating a number of elementary objects, including jobs, nodes, reservations, QoS structures, resource managers, and policies. Multiple minor elementary objects and composite objects are also used; these objects are defined in the [scheduling dictionary](#).

- [Jobs](#)
  - [Job States](#)
  - [Requirement \(or Req\)](#)
- [Nodes](#)
- [Advance Reservations](#)
- [Policies](#)
- [Resources](#)
- [Task](#)
- [PE](#)
- [Class \(or Queue\)](#)
- [Resource Manager \(RM\)](#)

Moab functions by manipulating a number of elementary objects, including jobs, nodes, reservations, QoS structures, resource managers, and policies. Multiple minor elementary objects and composite objects are also used; these objects are defined in the [scheduling dictionary](#).

## Jobs

Job information is provided to the Moab scheduler from a resource manager such as Loadleveler, PBS, Wiki, or LSF. Job attributes include ownership of the job, [job state](#), amount and type of resources required by the job, and a wallclock limit indicating how long the resources are required. A job consists of one or more [task groups](#), each of which requests a number of resources of a given type; for example, a job may consist of two task groups, the first asking for a single master task consisting of *1 IBM SP node with at least 512 MB of RAM* and the second asking for a set of slave tasks such as *24 IBM SP nodes with at least 128 MB of RAM*. Each task group consists of one or more [tasks](#) where a task is defined as the minimal independent unit of resources. By default, each task is equivalent to one processor. In SMP environments, however, users may wish to tie one or more processors together with a certain amount of memory and other resources.

## Job States

The job's *state* indicates its current status and eligibility for execution and can be any of the values listed in the following tables:

Table 3-1: Pre-execution states

| State            | Definition                                                                                                                                                                                                                                                                                     |
|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Deferred</b>  | Job that has been held by Moab due to an inability to schedule the job under current conditions. Deferred jobs are held for <a href="#">DEFERTIME</a> before being placed in the idle queue. This process is repeated <a href="#">DEFERCOUNT</a> times before the job is placed in batch hold. |
| <b>Hold</b>      | Job is idle and is not eligible to run due to a user, (system) administrator, or batch system <i>hold</i> (also, <a href="#">batchhold</a> , <a href="#">systemhold</a> , <a href="#">userhold</a> ).                                                                                          |
| <b>Idle</b>      | Job is currently queued and eligible to run but is not executing (also, <b>notqueued</b> ).                                                                                                                                                                                                    |
| <b>NotQueued</b> | The job has not been queued.                                                                                                                                                                                                                                                                   |
| <b>Unknown</b>   | Moab cannot determine the state of the job.                                                                                                                                                                                                                                                    |

Table 3-2: Execution states

| State            | Definition                                                                                                                                                                                            |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Starting</b>  | Batch system has attempted to start the job and the job is currently performing <i>pre-start</i> tasks that may include provisioning resources, staging data, or executing system pre-launch scripts. |
| <b>Running</b>   | Job is currently executing the user application.                                                                                                                                                      |
| <b>Suspended</b> | Job was running but has been suspended by the scheduler or an administrator; user application is still in place on the allocated compute resources, but it is not executing.                          |

Table 3-3: Post-execution states

| State            | Definition                                                                                                                                                                                                                                                           |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Completed</b> | Job has completed running without failure.                                                                                                                                                                                                                           |
| <b>Removed</b>   | Job has run to its requested walltime successfully but has been canceled by the scheduler or resource manager due to exceeding its walltime or violating another policy; includes jobs canceled by users or administrators either before or after a job has started. |

| State          | Definition                                                    |
|----------------|---------------------------------------------------------------|
| <b>Vacated</b> | Job canceled after partial execution due to a system failure. |

## Task Group (or Req)

A job *task group* (or req) consists of a request for a single type of resources. Each task group consists of the following components:

| Component                    | Description                                                                                                                                                                                |
|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Task Definition</b>       | A specification of the elementary resources that compose an individual task.                                                                                                               |
| <b>Resource Constraints</b>  | A specification of conditions that must be met for resource matching to occur. Only resources from nodes that meet <i>all</i> resource constraints may be allocated to the job task group. |
| <b>Task Count</b>            | The number of task instances required by the task group.                                                                                                                                   |
| <b>Task List</b>             | The list of nodes on which the task instances are located.                                                                                                                                 |
| <b>Task Group Statistics</b> | Statistics tracking resource utilization.                                                                                                                                                  |

## Nodes

Moab recognizes a node as a collection of resources with a particular set of associated attributes. This definition is similar to the traditional notion of a node found in a Linux cluster or supercomputer wherein a node is defined as one or more CPUs, associated memory, and possibly other compute resources such as local disk, swap, network adapters, and software licenses. Additionally, this node is described by various attributes such as an architecture type or operating system. Nodes range in size from small uniprocessor PCs to large symmetric multiprocessing (SMP) systems where a single node may consist of hundreds of CPUs and massive amounts of memory.

In many cluster environments, the primary source of information about the configuration and status of a compute node is the [resource manager](#). This information can be augmented by additional information sources including node monitors and information services. Further, extensive node policy and node configuration information can be specified within Moab via the graphical tools or the configuration file. Moab aggregates this information and presents a comprehensive view of the node configuration, usages, and state.

While a node in Moab in most cases represents a standard compute host, nodes may also be used to represent more generalized resources. The GLOBAL node possesses floating resources that are available cluster wide, and created virtual nodes (such as network, software, and data nodes) track and allocate resource usage for other resource types.

For additional node information, see [General Node Administration](#).

## Advance Reservations

An advance reservation dedicates a block of specific resources for a particular use. Each reservation consists of a list of resources, an access control list, and a time range for enforcing the access control list. The reservation ensures the matching nodes are used according to the access controls and policy constraints within the time frame specified. For example, a reservation could reserve 20 processors and 10 GB of memory for users Bob and John from Friday 6:00 a.m. to Saturday 10:00 p.m. Moab uses advance reservations extensively to manage backfill, guarantee resource availability for active jobs, allow service guarantees, support deadlines, and enable metascheduling. Moab also supports both regularly recurring reservations and the creation of dynamic one-time reservations for special needs. Advance reservations are described in detail in the [Advance Reservations](#) overview.

## Policies

A configuration file specifies policies controls how and when jobs start. Policies include job prioritization, fairness policies, fairshare configuration policies, and scheduling policies.

## Resources

Jobs, nodes, and reservations all deal with the abstract concept of a resource. A resource in the Moab world is one of the following:

| Resource          | Description                                             |
|-------------------|---------------------------------------------------------|
| <b>processors</b> | Specify with a simple count value                       |
| <b>memory</b>     | Specify real memory or RAM in megabytes (MB)            |
| <b>swap</b>       | Specify virtual memory or <i>swap</i> in megabytes (MB) |
| <b>disk</b>       | Specify local disk in megabytes (MB)                    |

In addition to these elementary resource types, there are two higher level resource concepts used within Moab: [Task](#) and the processor equivalent, or [\(PE\)](#).

## Task

A task is a collection of elementary resources that must be allocated together within a single [node](#). For example, a task may consist of one processor, 512 MB of RAM, and 2 GB of local disk. A key aspect of a task is that the resources associated with the task must be allocated as an atomic unit, without spanning node boundaries. A task requesting 2 processors cannot be satisfied by allocating 2 uniprocessor nodes, nor can a task requesting 1 processor and 1 GB of memory be satisfied by allocating 1 processor on 1 node and memory on another.

In Moab, when jobs or reservations request resources, they do so in terms of tasks typically using a task count and a task definition. By default, a task maps directly to a single processor within a job and maps to a full node within reservations. In all cases, this default definition can be overridden by specifying a new task definition.

Within both jobs and reservations, depending on task definition, it is possible to have multiple tasks from the same job mapped to the same node. For example, a job requesting 4 tasks using the default task definition of 1 processor per task, can be satisfied by 2 dual processor nodes.

## PE

The concept of the processor equivalent, or PE, arose out of the need to translate multi-resource consumption requests into a scalar value. It is not an elementary resource but rather a derived resource metric. It is a measure of the actual impact of a set of requested resources by a job on the total resources available system wide. It is calculated as follows:

$$PE = \text{MAX}(\text{ProcsRequestedByJob} / \text{TotalConfiguredProcs}, \text{MemoryRequestedByJob} / \text{TotalConfiguredMemory}, \text{DiskRequestedByJob} / \text{TotalConfiguredDisk}, \text{SwapRequestedByJob} / \text{TotalConfiguredSwap}) * \text{TotalConfiguredProcs}$$

For example, if a job requested 20% of the total processors and 50% of the total memory of a 128-processor MPP system, only two such jobs could be supported by this system. The job is essentially using 50% of all available resources since the system can only be scheduled to its most constrained resource - memory in this case. The processor equivalents for this job should be 50% of the processors, or PE = 64.

Another example: Assume a homogeneous 100-node system with 4 processors and 1 GB of memory per node. A job is submitted requesting 2 processors and 768 MB of memory. The PE for this job would be calculated as follows:

$$PE = \text{MAX}(2 / (100 * 4), 768 / (100 * 1024)) * (100 * 4) = 3.$$

This result makes sense since the job would be consuming 3/4 of the memory on a 4-processor node.

The calculation works equally well on homogeneous or heterogeneous systems, uniprocessor or large SMP systems.

## Class (or Queue)

A class (or queue) is a logical container object that implicitly or explicitly applies policies to jobs. In most cases, a class is defined and configured within the resource manager and associated with one or more of the following attributes or constraints:

| Attribute                     | Description                                                                                            |
|-------------------------------|--------------------------------------------------------------------------------------------------------|
| <b>Default Job Attributes</b> | A queue may be associated with a default job duration, default size, or default resource requirements. |

| Attribute               | Description                                                                                                                                               |
|-------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Host Constraints</b> | A queue may constrain job execution to a particular set of hosts.                                                                                         |
| <b>Job Constraints</b>  | A queue may constrain the attributes of jobs that may be submitted, including setting limits such as max wallclock time and minimum number of processors. |
| <b>Access List</b>      | A queue may constrain who may submit jobs into it based on such things as user lists and group lists.                                                     |
| <b>Special Access</b>   | A queue may associate special privileges with jobs including adjusted job priority.                                                                       |

As stated previously, most resource managers allow full class configuration within the resource manager. Where additional class configuration is required, the [CLASSCFG](#) parameter may be used.

Moab tracks class usage as a consumable resource allowing sites to limit the number of jobs using a particular class. This is done by monitoring class initiators that may be considered to be a ticket to run in a particular class. Any compute node may simultaneously support several types of classes and any number of initiators of each type. By default, nodes will have a one-to-one mapping between class initiators and configured processors. For every job task run on the node, one class initiator of the appropriate type is consumed. For example, a 3-processor job submitted to the class "batch" consumes three batch class initiators on the nodes where it runs.

Using queues as consumable resources allows sites to specify various policies by adjusting the class initiator to node mapping. For example, a site running serial jobs may want to allow a particular 8-processor node to run any combination of batch and special jobs subject to the following constraints:

- Only 8 jobs of any type allowed simultaneously.
- No more than 4 special jobs allowed simultaneously.

To enable this policy, the site may set the node's [MAXJOB](#) policy to 8 and configure the node with 4 special class initiators and 8 batch class initiators.

In virtually all cases, jobs have a one-to-one correspondence between processors requested and class initiators required. However, this is not a requirement, and with special configuration, sites may choose to associate job tasks with arbitrary combinations of class initiator requirements.

In displaying class initiator status, Moab signifies the type and number of class initiators available using the format [[<CLASSNAME>:<CLASSCOUNT>](#)]. This is most commonly seen in the output of node status commands indicating the number of configured and available class initiators, or in job status commands when displaying class initiator requirements.

## Resource Manager (RM)

While other systems may have more strict interpretations of a resource manager and its responsibilities, Moab's multi-resource manager support allows a much more liberal interpretation. In essence, any object that can provide environmental information and environmental control can be used

as a resource manager, including sources of resource, workload, credential, or policy information such as scripts, peer services, databases, web services, hardware monitors, or even flat files. Likewise, Moab considers to be a resource manager any tool that provides control over the cluster environment whether that be a license manager, queue manager, checkpoint facility, provisioning manager, network manager, or storage manager.

Moab aggregates information from multiple unrelated sources into a larger more complete world view of the cluster that includes all the information and control found within a standard resource manager such as [TORQUE](#), including node, job, and queue management services. For more information, see the [Resource Managers and Interfaces](#) overview.

### Arbitrary Resource

Nodes can also be configured to support various arbitrary resources. Use the [NODECFG](#) parameter to specify information about such resources. For example, you could configure a node to have *256 MB RAM, 4 processors, 1 GB Swap, and 2 tape drives*.

## Scheduling Dictionary

| Account                   |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b>         | A credential also known as "project ID." Multiple users may be associated a single account ID and each user may have access to multiple accounts. (See <a href="#">credential</a> definition and <a href="#">ACCOUNTCFG</a> parameter.)                                                                                                                                                                               |
| <b>Example</b>            | <code>ACCOUNT=hgc13</code>                                                                                                                                                                                                                                                                                                                                                                                            |
| ACL (Access Control List) |                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Definition</b>         | In the context of scheduling, an access control list is used and applied much as it is elsewhere. An ACL defines what credentials are required to access or use particular objects. The principal objects to which ACLs are applied are <a href="#">reservations</a> and <a href="#">QoSs</a> . ACLs may contain both allow and deny statements, include wildcards, and contain rules based on multiple object types. |
| <b>Example</b>            | Reservation META1 contains 4 access statements. <ul style="list-style-type: none"> <li>• Allow jobs owned by user "john" or "bob "</li> <li>• Allow jobs with QoS "premium"</li> <li>• Deny jobs in class "debug"</li> <li>• Allow jobs with a duration of less than 1 hour</li> </ul>                                                                                                                                |

| Allocation        |                                                                                                                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A logical, scalar unit assigned to users on a credential basis, providing access to a particular quantity of compute resources. Allocations are consumed by jobs associated with those credentials. |
| <b>Example</b>    | <code>ALLOCATION=30000</code>                                                                                                                                                                       |

| Class             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | (See <a href="#">Queue</a> ) A class is a logical container object that holds jobs allowing a site to associate various constraints and defaults to these jobs. Class access can also be tied to individual nodes defining whether a particular node will accept a job associated with a given class. Class based access to a node is denied unless explicitly allowed via resource manager configuration. Within Moab, classes are tied to jobs as a <a href="#">credential</a> . |
| <b>Example</b>    | job "cw.073" is submitted to class batch<br>node "cl02" accepts jobs in class batch<br>reservation weekend allows access to jobs in class batch                                                                                                                                                                                                                                                                                                                                    |

| CPU               |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A single processing unit. A CPU is a consumable resource. Nodes typically consist of one or more CPUs. (same as <a href="#">processor</a> ) |

| Credential        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | An attribute associated with <a href="#">jobs</a> and other objects that determines object identity. In the case of schedulers and resource managers, credential based policies and limits are often established. At submit time, jobs are associated with a number of credentials such as <a href="#">user</a> , <a href="#">group</a> , <a href="#">account</a> , <a href="#">QoS</a> , and <a href="#">class</a> . These job credentials subject the job to various polices and grant it various types of access. In most cases, credentials set both the privileges of the job and the ID of the actual job <a href="#">executable</a> . |
| <b>Example</b>    | Job "cw.24001" possesses the following credentials:<br><code>USER=john;GROUP=staff;ACCOUNT=[NONE];<br/>QOS=[DEFAULT];CLASS=batch</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

| Disk              |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A quantity of local disk available for use by batch jobs. Disk is a <a href="#">consumable resource</a> . |

## Execution Environment

**Definition** A description of the environment in which the executable is launched. This environment may include attributes such as the following:

- an executable
- command line arguments
- input file
- output file
- local user ID
- local group ID
- process resource limits

**Example** Job "cw.24001" possesses the following execution environment:

```
EXEC=/bin/sleep;ARGS="60";
INPUT=[NONE];OUTPUT=[NONE];
USER=loadl;GROUP=staff;
```

## Fairshare

**Definition** A mechanism that allows historical resource utilization information to be incorporated into job priority decisions.

## Fairness

**Definition** The access to shared compute resources that each user is granted. Access can be equal or based on factors such as historical resource usage, political issues, and job value.

## Group

**Definition** A [credential](#) typically directly mapping to a user's UNIX group ID.

## Job

**Definition** The fundamental object of resource consumption. A job contains the following components:

- A list of required [consumable resources](#)
- A list of [resource constraints](#) controlling which resources may be allocated to the job
- A list of [job constraints](#) controlling where, when, and how the job should run
- A list of [credentials](#)
- An [execution environment](#)

## Job Constraints

### Definition

A set of conditions that must be fulfilled for the job to start. These conditions are far reaching and may include one or more of the following:

- When the job may run. (After time X, within Y minutes.)
- Which resources may be allocated. (For example, node must possess at least 512 MB of RAM, run only in partition or Partition C, or run on HostA and HostB.)
- Starting job relative to a particular event. (Start after job X successfully completes.)

### Example

```
RELEASETIME>='Tue Feb 12, 11:00AM'
DEPEND=AFTERANY:cw.2004
NODEMEMORY==256MB
```

## Memory

### Definition

A quantity of physical memory (RAM). Memory is provided by compute nodes. It is required as a constraint or consumed as a consumable resource by jobs. Within Moab, memory is tracked and reported in megabytes (MB).

### Example

Node "node001" provides the following resources:

```
PROCS=1, MEMORY=512, SWAP=1024
```

"Job cw.24004" consumes the following resources per task:

```
PROCS=1, MEMORY=256
```

## Node

### Definition

A node is the fundamental object associated with compute resources. Each node contains the following components:

- A list of [consumable resources](#)
- A list of [node attributes](#)

## Node Attribute

### Definition

A node attribute is a non-quantitative aspect of a node. Attributes typically describe the node itself or possibly aspects of various node resources such as processors or memory. While it is probably not optimal to aggregate node and resource attributes together in this manner, it is common practice. Common node attributes include processor architecture, operating system, and processor speed. Jobs often specify that resources be allocated from nodes possessing certain node attributes.

### Example

```
ARCH=AMD, OS=LINUX24, PROCSPEED=950
```

| Node Feature      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A node feature is a <a href="#">node attribute</a> that is typically specified locally via a configuration file. Node features are opaque strings associated with the node by the resource manager that generally only have meaning to the end-user, or possibly to the scheduler. A node feature is commonly associated with a subset of nodes allowing end-users to request use of this subset by requiring that resources be allocated from nodes with this feature present. In many cases, node features are used to extend the information provided by the resource manager. |
| <b>Example</b>    | <pre>FEATURE=s950,pIII,geology</pre> <p><i>This may be used to indicate that the node possesses a 950 MHz Pentium III processor and that the node is owned by the Geology department.</i></p>                                                                                                                                                                                                                                                                                                                                                                                     |

| Processor         |                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A processing unit. A processor is a consumable resource. Nodes typically consist of one or more processors. (same as CPU) |

| Quality of Service (QoS) |                                                                    |
|--------------------------|--------------------------------------------------------------------|
| <b>Definition</b>        | An object that provides special services, resources, and so forth. |

| Queue             |                              |
|-------------------|------------------------------|
| <b>Definition</b> | (see <a href="#">Class</a> ) |

| Reservation       |                                                                                                                                    |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | An object that reserves a specific collection or resources for a specific timeframe for use by jobs that meet specific conditions. |
| <b>Example</b>    | Reserve 24 processors and 8 GB of memory from time T1 to time T2 for use by user X or jobs in the class batch.                     |

| Resource          |                                                                                                                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | Hardware, generic resources such as software, and features available on a node, including memory, disk, swap, and processors. |

## Resource, Available

### Definition

A compute node's [configured](#) resources minus the *maximum* of the sum of the resources [utilized](#) by all job tasks running on the node and the resources [dedicated](#); that is,  $R_{Available} = R_{Configure} - \text{MAX}(R_{Dedicated}, R_{Utilized})$ .

In most cases, resources utilized will be associated with compute jobs that the batch system has started on the compute nodes, although resource consumption may also come from the operating system or *rogue* processes outside of the batch system's knowledge or control. Further, in a well-managed system, utilized resources are less than or equal to dedicated resources and when exceptions are detected, one or more [usage-based limits](#) are activated to [preempt](#) the jobs violating their requested resource usage.

### Example

Node "cl003" has 4 processors and 512 MB of memory. It is executing 2 tasks of job "clserver.0041" that are using 1 processor and 60 MB of memory each. One processor and 250 MB of memory are reserved for user "jsmith" but are not currently in use.

Resources available to user jsmith on node "cl003":

- 2 processors
- 392 MB memory

Resources available to a user other than jsmith on node "cl003":

- 1 processor
- 142 MB memory

## Resource, Configured

### Definition

The total amount of [consumable resources](#) that are available on a compute node for use by job tasks.

### Example

Node "cl003" has 4 processors and 512 MB of memory. It is executing 2 tasks of job "clserver.0041" that are using 1 processor and 60 MB of memory each. One processor and 250 MB of memory are reserved for user "jsmith" but are not currently in use.

Configured resources for node "cl003":

- 4 processors
- 512 MB memory

### Resource, Consumable

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | <p>Any object that can be used (that is, consumed and thus made unavailable to another job) by, or dedicated to a job is considered to be a resource. Common examples of resources are a node's physical memory or local disk. As these resources may be given to one job and thus become unavailable to another, they are considered to be consumable. Other aspects of a node, such as its operating system, are not considered to be consumable since its use by one job does not preclude its use by another. Note that some node objects, such as a network adapter, may be dedicated under some operating systems and resource managers and not under others. On systems where the network adapter cannot be dedicated and the network usage per job cannot be specified or tracked, network adapters are not considered to be resources, but rather attributes.</p> <p>Nodes possess a specific quantity of consumable resources such as real memory, local disk, or processors. In a resource management system, the node manager may choose to report only those configured resources available to batch jobs. For example, a node may possess an 80-GB hard drive but may have only 20 GB dedicated to batch jobs. Consequently, the resource manager may report that the node has 20 GB of local disk available when idle. Jobs may explicitly request a certain quantity of consumable resources.</p> |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### Resource, Constraint

|                   |                                                                                                                                                                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | <p>A resource constraint imposes a rule on which resources can be used to match a resource request. Resource constraints either specify a required quantity and type of resource or a required node attribute. All resource constraints must be met by any given node to establish a match.</p> |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### Resource, Dedicated

|                   |                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | <p>A job may request that a block of resources be dedicated while the job is executing. At other times, a certain number of resources may be reserved for use by a particular user or group. In these cases, the scheduler is responsible for guaranteeing that these resources, <a href="#">utilized</a> or not, are set aside and made unavailable to other jobs.</p>                              |
| <b>Example</b>    | <p>Node "cl003" has 4 processors and 512 MB of memory. It is executing 2 tasks of job "clserver.0041" that are using 1 processor and 60 MB of memory each. One processor and 250 MB of memory are reserved for user "jsmith" but are not currently in use.</p> <p>Dedicated resources for node "cl003":</p> <ul style="list-style-type: none"> <li>• 1 processor</li> <li>• 250 MB memory</li> </ul> |

### Resource, Utilized

|                   |                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | <p>All <a href="#">consumable</a> resources actually used by all job tasks running on the compute node.</p> |
|-------------------|-------------------------------------------------------------------------------------------------------------|

## Resource, Utilized

|                |                                                                                                                                                                                                                                                                                                                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Example</b> | <p>Node "cl003" has 4 processors and 512 MB of memory. It is executing 2 tasks of job "clserver.0041" that are using 1 processor and 60 MB of memory each. One processor and 250 MB of memory are reserved for user "jsmith" but are not currently in use.</p> <p>Utilized resources for node "cl003":</p> <ul style="list-style-type: none"> <li>• 2 processors</li> <li>• 120 MB memory</li> </ul> |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Swap

|                   |                                                                                                                                     |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | A quantity of virtual memory available for use by batch jobs. Swap is a consumable resource provided by nodes and consumed by jobs. |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------|

## Task

|                   |                                               |
|-------------------|-----------------------------------------------|
| <b>Definition</b> | An atomic collection of consumable resources. |
|-------------------|-----------------------------------------------|

## User, Global

|                   |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Definition</b> | The user credential used to provide access to functions and resources. In local scheduling, global user IDs map directly to local user IDs. |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|

## User, Local

|                   |                                                                      |
|-------------------|----------------------------------------------------------------------|
| <b>Definition</b> | The user credential under which the job executable will be launched. |
|-------------------|----------------------------------------------------------------------|

## Workload

|                   |                   |
|-------------------|-------------------|
| <b>Definition</b> | Generalized term. |
|-------------------|-------------------|

# Scheduling Iterations and Job Flow

- [Scheduling Iterations](#)
  - [Update State Information](#)
  - [Handle User Requests](#)

- [Perform Next Scheduling Cycle](#)
- [Detailed Job Flow](#)
  - [Determine Basic Job Feasibility](#)
  - [Prioritize Jobs](#)
  - [Enforce Configured Throttling Policies](#)
  - [Determine Resource Availability](#)
  - [Allocate Resources to Job](#)
  - [Launch Job](#)

## Scheduling Iterations

In any given scheduling iteration, many activities take place, examples of which are listed below:

- [Refresh reservations](#)
- [Schedule reserved jobs](#)
- [Schedule priority jobs](#)
- [Backfill jobs](#)
- [Update statistics](#)
- [Update State Information](#)
- [Handle User Requests](#)
- [Perform Next Scheduling Cycle](#)

### Update State Information

Each iteration, the scheduler contacts the resource manager(s) and requests up-to-date information on compute resources, workload, and policy configuration. On most systems, these calls are to a centralized resource manager daemon that possesses all information. Jobs may be reported as being in any of the following states listed in the [job state](#) table.

### Handle User Requests

User requests include any call requesting state information, configuration changes, or job or resource manipulation commands. These requests may come in the form of user client calls, peer daemon calls, or process signals.

### Perform Next Scheduling Cycle

Moab operates on a polling/event driven basis. When all scheduling activities complete, Moab processes user requests until a new resource manager event is received or an internal event is generated. Resource manager events include activities such as a new job submission or completion of an active job, addition of new node resources, or changes in resource manager policies. Internal events include

administrator [schedule](#) requests, reservation activation/deactivation, or the expiration of the [RMPOLLINTERVAL](#) timer.

## Detailed Job Flow

### Determine Basic Job Feasibility

The first step in scheduling is determining which jobs are feasible. This step eliminates jobs that have job holds in place, invalid job states (such as Completed, Not Queued, Deferred), or unsatisfied preconditions. Preconditions may include stage-in files or completion of preliminary job steps.

### Prioritize Jobs

With a list of feasible jobs created, the next step involves [determining the relative priority](#) of all jobs within that list. A priority for each job is calculated based on job attributes such as job owner, job size, and length of time the job has been queued.

### Enforce Configured Throttling Policies

Any configured [throttling policies](#) are then applied constraining how many jobs, nodes, processors, and so forth are allowed on a per credential basis. Jobs that violate these policies are not considered for scheduling.

### Determine Resource Availability

For each job, Moab attempts to locate the required compute resources needed by the job. For a match to be made, the node must possess all node attributes specified by the job and possess adequate available resources to meet the "TasksPerNode" job constraint. (Default "TasksPerNode" is 1.) Normally, Moab determines that a node has adequate resources if the resources are *neither utilized by nor dedicated to* another job using the calculation.

$R.Available = R.Configured - MAX(R.Dedicated, R.Utilized)$ .

The [NODEAVAILABILITYPOLICY on page 1000](#) parameter can be modified to adjust this behavior.

### Allocate Resources to Job

If adequate resources can be found for a job, the [node allocation policy](#) is then applied to select the best set of resources. These allocation policies allow selection criteria such as speed of node, type of reservations, or excess node resources to be figured into the allocation decision to improve the performance of the job and maximize the freedom of the scheduler in making future scheduling decisions.

### Launch Job

With the resources selected and task distribution mapped, the scheduler then contacts the resource manager and informs it where and how to launch the job. The resource manager then initiates the actual job executable.

## Configuring the Scheduler

- [Adjusting Server Behavior](#)
  - [Logging](#)
  - [Checkpointing](#)
  - [Client Interface](#)
  - [Scheduler Mode](#)
  - [Configuring a job ID offset on page 116](#)

Scheduler configuration is maintained using the flat text configuration file `moab.cfg`. All configuration file entries consist of simple `<PARAMETER> <VALUE>` pairs that are whitespace delimited. Parameter names are not case sensitive but `<VALUE>` settings are. Some parameters are array values and should be specified as `<PARAMETER>[<INDEX>]` (Example: `QOSCFG[hiprio] PRIORITY=1000`); the `<VALUE>` settings may be integers, floats, strings, or arrays of these. Some parameters can be specified as arrays wherein index values can be numeric or alphanumeric strings. If no array index is specified for an array parameter, an index of zero (0) is assumed. The example below includes both array based and non-array based parameters:

```
SCHEDCFG[cluster2] SERVER=head.c2.org MODE=NORMAL
LOGLEVEL 6
LOGDIR /var/tmp/moablog
```

See the [parameters](#) documentation for information on specific parameters.

The `moab.cfg` file is read when Moab is started up or recycled. Also, the `mschedctl -m` command can be used to reconfigure the scheduler at any time, updating some or all of the configurable parameters dynamically. This command can be used to modify parameters either permanently or temporarily. For example, the command `mschedctl -m LOGLEVEL 3` will temporarily adjust the scheduler log level. When the scheduler restarts, the log level restores to the value stored in the Moab configuration files. To adjust a parameter permanently, the option `--flags=persistent` should be set.

At any time, the current server parameter settings may be viewed using the `mschedctl -l` command.

### Adjusting Server Behavior

Most aspects of Moab behavior are configurable. This includes both scheduling policy behavior and daemon behavior. In terms of configuring server behavior, the following realms are most commonly modified.

#### Logging

Moab provides extensive and highly configurable logging facilities controlled by parameters.

| Parameter              | Description                        |
|------------------------|------------------------------------|
| <a href="#">LOGDIR</a> | Indicates directory for log files. |

| Parameter                               | Description                                        |
|-----------------------------------------|----------------------------------------------------|
| <a href="#"><u>LOGFACILITY</u></a>      | Indicates scheduling facilities to track.          |
| <a href="#"><u>LOGFILE</u></a>          | Indicates path name of log file.                   |
| <a href="#"><u>LOGFILEMAXSIZE</u></a>   | Indicates maximum size of log file before rolling. |
| <a href="#"><u>LOGFILEROLLDEPTH</u></a> | Indicates maximum number of log files to maintain. |
| <a href="#"><u>LOGLEVEL</u></a>         | Indicates verbosity of logging.                    |

## Checkpointing

Moab checkpoints its internal state. The checkpoint file records statistics and attributes for jobs, nodes, reservations, users, groups, classes, and almost every other scheduling object.

| Parameter                                       | Description                                                                                                                                        |
|-------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#"><u>CHECKPOINTEXPIRATIONTIME</u></a> | Indicates how long unmodified data should be kept after the associated object has disappeared; that is, job priority for a job no longer detected. |
| <a href="#"><u>CHECKPOINTFILE</u></a>           | Indicates path name of checkpoint file.                                                                                                            |
| <a href="#"><u>CHECKPOINTINTERVAL</u></a>       | Indicates interval between subsequent checkpoints.                                                                                                 |

## Client Interface

The Client interface is configured using the [SCHEDCFG](#) parameter. Most commonly, the attributes **SERVER** and **PORT** must be set to point client commands to the appropriate Moab server. Other parameters such as [CLIENTTIMEOUT](#) may also be set.

## Scheduler Mode

The scheduler mode of operation is controlled by setting the **MODE** attribute of the [SCHEDCFG](#) parameter. The following modes are allowed:

| Mode               | Description                                                                                                                                              |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>INTERACTIVE</b> | Moab interactively confirms each scheduling action before taking any steps. (See <a href="#"><u>interactive mode overview</u></a> for more information.) |

| Mode              | Description                                                                                                                                                                                                                                                                                                                                        |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MONITOR</b>    | Moab observes cluster and workload performance, collects statistics, interacts with allocation management services, and evaluates failures, but it does not actively alter the cluster, including job migration, workload scheduling, and resource provisioning. (See <a href="#">monitor mode overview</a> for more information.)                 |
| <b>NORMAL</b>     | Moab actively schedules workload according to mission objectives and policies; it creates reservations; starts, cancels, preempts, and modifies jobs; and takes other scheduling actions.                                                                                                                                                          |
| <b>SIMULATION</b> | Moab obtains <a href="#">workload</a> and <a href="#">resource</a> information from specified simulation trace files and schedules the defined virtual environment.                                                                                                                                                                                |
| <b>SINGLESTEP</b> | Moab behaves as in <a href="#">NORMAL</a> mode but will only schedule a single iteration and then exit.                                                                                                                                                                                                                                            |
| <b>SLAVE</b>      | Moab behaves as in <a href="#">NORMAL</a> mode but will only start a job when explicitly requested by a trusted <a href="#">grid peer</a> service or <a href="#">administrator</a> .                                                                                                                                                               |
| <b>TEST</b>       | Moab behaves as in <a href="#">NORMAL</a> mode, will make reservations, and scheduling decisions, but will then only log scheduling actions it would have taken if running in <a href="#">NORMAL</a> mode. In most cases, "TEST" mode is identical to <a href="#">MONITOR</a> mode. (See <a href="#">test mode overview</a> for more information.) |

## Configuring a job ID offset

Moab assigns job IDs as integers in numeric order as jobs are submitted, starting with 1. In some situations, you might want to offset the integer at which Moab starts to assign job IDs in your system.

This example describes how you would offset the job IDs in a compound system consisting of Site A, Site B, and Site C, each of which runs its own instance of Moab. Users belonging to any of the sites can submit jobs to their own site and to the other two. To simplify aggregation of usage records from the three sites, offset the job IDs for Site B to a starting value higher than the expected total lifetime value for the system; in this example, to **20000000**. Likewise, set Site C to 20,000,000 more, or **40000000**. To do so, set the **MINJOBID** attribute of **SCHEDCFG** in each system's `moab.cfg` to the offset value. To ensure that Moab will never use the same job ID for two different sites, also set **MAXJOBID**. If the Moab job naming process ever reaches the **MAXJOBID**, it will start over again with the **MINJOBID**.

```
SCHEDCFG[moab] SERVER=moab_siteA:4244 MAXJOBID=19999999
```

```
SCHEDCFG[moab] SERVER=moab_siteB:4344 MINJOBID=20000000 MAXJOBID=39999999
```

```
SCHEDCFG[moab] SERVER=moab_siteC:4444 MINJOBID=40000000 MAXJOBID=59999999
```

When users submit jobs to Moab using [msub on page 286](#), Moab selects the job ID in numeric order, starting with 1 in Site A, 20000000 in Site B, and 40000000 in Site C.

If the compound system in this example uses TORQUE as its resource manager and users submit jobs directly to TORQUE using [qsub](#), TORQUE assigns the job ID instead of Moab. In this case, you should also

offset the TORQUE job IDs by setting the [next\\_job\\_number on page 2779](#) server parameter of Site B and Site C to `20000000` and `40000000`, respectively.

```
$user qmgr "set server next_job_number=20000000"
```

```
$user qmgr "set server next_job_number=40000000"
```

**i** TORQUE job ID limits will allow you to use the 20,000,000 offset scheme for up to 4 sites.

#### Related topics

- [Initial Configuration](#)
- Adding `#INCLUDE` files to `moab.cfg`

## Credential Overview

Moab supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features. There are five types of credentials - [user](#), [group](#), [account](#), [class](#), and [QoS](#). While the credentials have many similarities, each plays a slightly different role.

- [General Credential Attributes](#)
- [User Credential](#)
- [Group Credential](#)
- [Account \(or Project\) Credential](#)
- [Class \(or Queue\) Credential](#)
- [QoS Credential](#)

## General Credential Attributes

Internally, credentials are maintained as objects. Credentials can be created, destroyed, queried, and modified. They are associated with jobs and requests providing access and privileges. Each credential type has the following attributes:

- [Priority Settings](#)
- [Usage Limits](#)
- [Service Targets](#)
- [Credential and Partition Access](#)
- [Statistics](#)
- [Credential Defaults, State and Configuration Information](#)

All credentials represent a form of identity, and when applied to a job, express ownership. Consequently, jobs are subject to policies and limits associated with their owners.

## Credential Priority Settings

Each credential may be assigned a priority using the **PRIORITY** attribute. This priority affects a job's total credential priority factor as described in the [Priority Factors](#) section. In addition, each credential may also specify priority weight offsets, which adjust priority weights that apply to associated jobs. These priority weight offsets include [FSWEIGHT](#) (See [Priority-Based Fairshare](#) for more information.), [QTWEIGHT](#), and [XFWEIGHT](#).

For example:

```
# set priority weights
CREDWEIGHT 1
USERWEIGHT 1
CLASSWEIGHT 1
SERVICEWEIGHT 1
XFACTORWEIGHT 10
QUEUETIMEWEIGHT 1000
# set credential priorities
USERCFG[john] PRIORITY=200
CLASSCFG[batch] PRIORITY=15
CLASSCFG[debug] PRIORITY=100
QOFCFG[bottomfeeder] QTWEIGHT=-50 XFWEIGHT=100
ACCOUNTCFG[topfeeder] PRIORITY=100
```

## Credential Usage Limits

Usage limits constrain which jobs may run, which jobs may be considered for scheduling, and what quantity of resources each individual job may consume. With usage limits, policies such as [MAXJOB](#), [MAXNODE](#), and [MAXMEM](#) may be enforced against both idle and active jobs. Limits may be applied in any combination as shown in the example below where usage limits include 32 active processors per group and 12 active jobs for user *john*. For a job to run, it must satisfy the most limiting policies of all associated credentials. The [Throttling Policy](#) section documents credential usage limits in detail.

```
GROUPCFG[DEFAULT] MAXPROC=32 MAXNODE=100
GROUPCFG[staff] MAXNODE=200
USERCFG[john] MAXJOB=12
```

## Service Targets

Credential service targets allow jobs to obtain special treatment to meet usage or response time based metrics. Additional information about service targets can be found in the [Fairshare](#) section.

## Credential and Partition Access

Access to partitions and to other credentials may be specified on a per credential basis with credential [access lists](#), [default credentials](#), and credential [membership lists](#).

### Credential Access Lists

You can use the **ALIST**, **PLIST**, and **QLIST** attributes (shown in the following table) to specify the list of credentials or partitions that a given credential may access.

| Credential       | Attribute                                                                           |
|------------------|-------------------------------------------------------------------------------------|
| <b>Account</b>   | <b>ALIST</b> (allows credential to access specified list of accounts)               |
| <b>Partition</b> | <b>PLIST</b> (allows credential to access specified list of partitions)             |
| <b>QoS</b>       | <b>QLIST</b> (allows credential to access specified list of <a href="#">QoSes</a> ) |

*Example 3-1:*

```
USERCFG[bob]    ALIST=jupiter,quantum
USERCFG[steve] ALIST=quantum
```

**i** Account-based access lists are only enforced if using an allocation manager or if the **ENFORCEACCOUNTACCESS** parameter is set to "TRUE."

*Assigning Default Credentials*

Use the **\*DEF** attribute (shown in the following table) to specify the default credential or partition for a particular credential.

| Credential     | Attribute                               |
|----------------|-----------------------------------------|
| <b>Account</b> | <b>ADEF</b> (specifies default account) |
| <b>Class</b>   | <b>CDEF</b> (specifies default class)   |
| <b>QoS</b>     | <b>QDEF</b> (specifies default QoS)     |

*Example 3-2:*

```
# user bob can access accounts a2, a3, and a6. If no account is explicitly requested,
# his job will be assigned to account a3
USERCFG[bob]    ALIST=a2,a3,a6 ADEF=a3
# user steve can access accounts a14, a7, a2, a6, and a1. If no account is explicitly
# requested, his job will be assigned to account a2
USERCFG[steve] ALIST=a14,a7,a2,a6,a1 ADEF=a2
```

*Specifying Credential Membership Lists*

As an alternate to specifying access lists, administrators may also specify membership lists. This allows a credential to specify who can access it rather than allowing each credential to specify which credentials it can access. Membership lists are controlled using the **MEMBERULIST**, **EXCLUDEUSERLIST** and **REQUIREDUSERLIST** attributes, shown in the following table:

| Credential          | Attribute                                                            |
|---------------------|----------------------------------------------------------------------|
| User                | ---                                                                  |
| Account, Group, QoS | <b>MEMBERULIST</b>                                                   |
| Class               | <a href="#">EXCLUDEUSERLIST</a> and <a href="#">REQUIREDUSERLIST</a> |

Example 3-3:

```
# account omega3 can only be accessed by users johnh, stevek, jenp
ACCOUNTCFG[omega3] MEMBERULIST=johnh,stevek,jenp
```

Example 3-4: Controlling Partition Access on a Per User Basis

A site may specify the user john may access partitions atlas, pluto, and zeus and will default to partition pluto. To do this, include the following line in the configuration file:

```
USERCFG[john] PLIST=atlas,pluto,zeus
```

Example 3-5: Controlling QoS Access on a Per Group Basis

A site may also choose to allow everyone in the group staff to access QoS standard and special with a default QoS of standard. To do this, include the following line in the configuration file:

```
GROUPCFG[staff] QLIST=standard,special QDEF=standard
```

Example 3-6: Controlling Resource Access on a Per Account Basis

An organization wants to allow everyone in the account omega3 to access nodes 20 through 24. To do this, include the following in the configuration file:

```
ACCOUNTCFG[omega3] MEMBERULIST=johnh,stevek,jenp
SRCFG[omega3] HOSTLIST=r:20-24 ACCOUNTLIST=omega3
```

## Credential Statistics

Full statistics are maintained for each credential instance. These statistics record current and historical resource usage, level of service delivered, accuracy of requests, and many other aspects of workload. Note, though, that you must explicitly enable credential statistics as they are not tracked by default. You can enable credential statistics by including the following in the configuration file:

```
USERCFG[DEFAULT]          ENABLEPROFILING=TRUE
GROUPCFG[DEFAULT]        ENABLEPROFILING=TRUE
ACCOUNTCFG[DEFAULT]      ENABLEPROFILING=TRUE
CLASSCFG[DEFAULT]        ENABLEPROFILING=TRUE
QOSCFG[DEFAULT]          ENABLEPROFILING=TRUE
```

## Job Defaults, Credential State, and General Configuration

Credentials may apply defaults and force job configuration settings via the following parameters:

| COMMENT            |                                                                                                                                   |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b> | Associates a comment string with the target credential.                                                                           |
| <b>Example</b>     | <pre>USERCFG[steve] COMMENT='works for boss, provides good service' CLASSCFG[i3] COMMENT='queue for I/O intensive workload'</pre> |

| HOLD               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b> | <p>Specifies a hold should be placed on all jobs associated with the target credential.</p> <div style="border: 1px solid black; padding: 5px; margin: 5px 0;"> <p><b>i</b> The order in which this <b>HOLD</b> attribute is evaluated depends on the following credential precedence: <b>USERCFG</b>, <b>GROUPCFG</b>, <b>ACCOUNTCFG</b>, <b>CLASSCFG</b>, <b>QOSCFG</b>, <b>USERCFG</b> <i>[DEFAULT]</i>, <b>GROUPCFG</b> <i>[DEFAULT]</i>, <b>ACCOUNTCFG</b> <i>[DEFAULT]</i>, <b>CLASSCFG</b> <i>[DEFAULT]</i>, <b>QOSCFG</b> <i>[DEFAULT]</i>.</p> </div> |
| <b>Example</b>     | <pre>GROUPCFG[bert] HOLD=yes</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

| JOBFLAGS           |                                                                                            |
|--------------------|--------------------------------------------------------------------------------------------|
| <b>Description</b> | Assigns the specified <a href="#">job flag</a> to all jobs with the associated credential. |
| <b>Example</b>     | <pre>CLASSCFG[batch] JOBFLAGS=suspendable QOSCFG[special] JOBFLAGS=restartable</pre>       |

| NOSUBMIT           |                                                                                               |
|--------------------|-----------------------------------------------------------------------------------------------|
| <b>Description</b> | Specifies whether jobs belonging to this credential can submit jobs using <code>msub</code> . |
| <b>Example</b>     | <pre>ACCOUNTCFG[general] NOSUBMIT=TRUE CLASSCFG[special] NOSUBMIT=TRUE</pre>                  |

| OVERRUN            |                                                                                                                                          |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b> | Specifies the amount of time a job may exceed its wallclock limit before being terminated. (Only applies to user and class credentials.) |
| <b>Example</b>     | <pre>CLASSCFG[bigmem] OVERRUN=00:15:00</pre>                                                                                             |

| VARIABLE           |                                                                                                                                                                   |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b> | Specifies attribute-value pairs associated with the specified credential. These variables may be used in triggers and other interfaces to modify system behavior. |
| <b>Example</b>     | <pre>GROUPCFG[staff] VARIABLE='nocharge=true'</pre>                                                                                                               |

Credentials may carry additional configuration information. They may specify that detailed statistical profiling should occur, that submitted jobs should be held, or that corresponding jobs should be marked as preemptible.

## User Credential

The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

As the fundamental credential, the user credential has a number of unique attributes.

- [Role](#)
- [Email Address](#)
- [Disable Moab User Email](#)

### Role

Moab supports role-based authorization, mapping particular roles to collections of specific users. See the [Security](#) section for more information.

### Email Address

Facilities exist to allow user notification in the event of job or system failures or under other general conditions. This attribute allows these notifications to be mailed directly to the target user.

```
USERCFG[sally] EMAILADDRESS=sally@acme.com
```

### Disable Moab User Email

You can disable Moab email notifications for a specific user.

```
USERCFG[john] NOEMAIL=TRUE
```

## Group Credential

The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-

to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.

With many resource managers such as TORQUE, PBSPro, and LSF, the group associated with a job is either the user's active primary group as specified within the operating system or a group that is explicitly requested at job submission time. When a secondary group is requested, the user's default group and associated policies are not taken into account. Also note that a job may only run under one group. If more constraining policies are required for these systems, an alternate aggregation scheme such as the use of [Account](#) or [QOS](#) credentials is recommended.

To submit a job as a secondary group, refer to your local resource manager's job submission options. For TORQUE users, see the `group_list=g_list` option of the [qsub -W](#) command.

## Account Credential

The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes. User-to-accounting mapping may be obtained from a resource manager or allocation manager, or you can configure it directly within Moab. Access to an account can be controlled via the **ALIST** and **ADEF** credential attributes specified via the [Identity Manager](#) or the `moab.cfg` file.

The **MANAGERS** attribute (applicable only to the account and [class](#) credentials) allows an administrator to assign a user the ability to manage jobs inside the credential, as if the user is the job owner.

### Example 3-7: MANAGERS Attribute

```
ACCOUNTCFG [general]   MANAGERS=ops
ACCOUNTCFG [special]  MANAGERS=stevep
```

If a user is able to access more than one account, the desired account can be specified at job submission time using the resource-manager specific attribute. For example, with [TORQUE](#) this is accomplished using the `-A` argument to the [qsub](#) command.

### Example 3-8: Enforcing Account Usage

Job-to-account mapping can be enforced using the **ALIST** attribute and the [ENFORCEACCOUNTACCESS](#) parameter.

```
USERCFG [john]        ALIST=proj1,proj3
USERCFG [steve]       ALIST=proj2,proj3,proj4
USERCFG [brad]        ALIST=proj1
USERCFG [DEFAULT]    ALIST=proj2
ENFORCEACCOUNTACCESS TRUE
...
```

## Class Credential

- [Class Job Defaults](#)
- [Per Job Min/Max Limits](#)
- [Resource Access](#)
- [Class Membership Constraints](#)

- [Attributes Enabling Class Access to Other Credentials](#)
- [Special Class Attributes \(such as Managers and Job Prologs\)](#)
- [Setting Default Classes](#)
- [Creating a Remap Class](#)
- [Class Attribute Overview](#)
- [Enabling Queue Complex Functionality](#)

The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab. Class attributes include the following:

- [Job Defaults](#)
- [Per Job Min/Max Limits](#)
- [Resource Access Constraints](#)
- [Class Membership Constraints](#)
- [Attributes Enabling Class Access to Other Credentials](#)
- [Special Class Attributes](#)

**i** When using [SLURM](#), Moab classes have a one-to-one relationship with SLURM partitions of the same name.

**i** For all classes configured in Moab, a resource manager queue with the same name should be created.

**i** When TORQUE reports a new queue to Moab a class of the same name is automatically applied to all nodes.

### Class Job Defaults

Classes can be assigned to a default [job template](#) that can apply values to job attributes not explicitly specified by the submitter. Additionally, you can specify shortcut attributes from the table that follows:

| Attribute           | Description           |
|---------------------|-----------------------|
| <b>DEFAULT.ATTR</b> | Job Attribute         |
| <b>DEFAULT.DISK</b> | Required Disk (in MB) |

| Attribute               | Description                           |
|-------------------------|---------------------------------------|
| <b>DEFAULT.EXT</b>      | Job RM Extension                      |
| <b>DEFAULT.FEATURES</b> | Required Node Features/Properties     |
| <b>DEFAULT.GRES</b>     | Required Consumable Generic Resources |
| <b>DEFAULT.MEM</b>      | Required Memory/RAM (in MB)           |
| <b>DEFAULT.NODESET</b>  | Node Set Specification                |
| <b>DEFAULT.PROC</b>     | Required Processor Count              |
| <b>DEFAULT.TPN</b>      | Tasks Per Node                        |
| <b>DEFAULT.WCLIMIT</b>  | Wallclock Limit                       |

**i** Defaults set in a class/queue of the resource manager will override the default values of the corresponding class/queue specified in Moab.

**i** [RESOURCELIMITPOLICY](#) must be configured in order for the **CLASSCFG** limits to take effect.

Example 3-9:

```
CLASSCFG [batch] DEFAULT.DISK=200MB DEFAULT.FEATURES=prod DEFAULT.WCLIMIT=1:00:00
CLASSCFG [debug] DEFAULT.FEATURES=debug DEFAULT.WCLIMIT=00:05:00
```

### Per Job Min/Max Limits

Classes can be assigned a minimum and a maximum [job template](#) that constrains resource requests. Jobs submitted to a particular queue must meet the resource request constraints of these templates. If a job submission exceeds these limits, the entire job submission fails.

| Limit                   | Description                   |
|-------------------------|-------------------------------|
| <b>MAX.ARRAYSUBJOBS</b> | Max Allowed Jobs in an Array  |
| <b>MAX.CPUTIME</b>      | Max Allowed Utilized CPU Time |
| <b>MAX.NODE</b>         | Max Allowed Node Count        |

| Limit              | Description                     |
|--------------------|---------------------------------|
| <b>MAX.PROC</b>    | Max Allowed Processor Count     |
| <b>MAX.PS</b>      | Max Requested Processor-Seconds |
| <b>MIN.NODE</b>    | Min Allowed Node Count          |
| <b>MIN.PROC</b>    | Min Allowed Processor Count     |
| <b>MIN.PS</b>      | Min Requested Processor-Seconds |
| <b>MIN.TPN</b>     | Min Tasks Per Node              |
| <b>MIN.WCLIMIT</b> | Min Requested Wallclock Limit   |
| <b>MAX.WCLIMIT</b> | Max Requested Wallclock Limit   |

**i** The parameters listed in the preceding table are for classes and PARCFG only, not users, accounts, groups or QoSes, and they function on a per-job basis. The **MAX.\*** and **MIN.\*** parameters are different from the **MAXJOB**, **MAXNODE**, and **MAXMEM** parameters described earlier in [Credential Usage Limits](#).

## Resource Access

Classes may be associated with a particular set of compute resources. Consequently, jobs submitted to a given class may only use listed resources. This may be handled at the [resource manager](#) level or via the [CLASSCFG HOSTLIST](#) attribute.

## Class Membership Constraints

Classes may be configured at either the resource manager or scheduler level to only allow select users and groups to access them. Jobs that do not meet these criteria are rejected. If specifying class membership/access at the resource manager level, see the respective resource manager documentation. Moab automatically detects and enforces these constraints. If specifying class membership/access at the scheduler level, use the **REQUIREDUSERLIST** or **EXCLUDEUSERLIST** attributes of the [CLASSCFG](#) parameter.

**i** Under most resource managers, jobs must always be a member of one and only one class.

## Attributes Enabling Class Access to Other Credentials

Classes may be configured to allow jobs to access other credentials such as QoSs and Accounts. This is accomplished using the [QDEF](#), [QLIST](#), [ADEF](#), and [ALIST](#) attributes.

## Special Class Attributes

The class object also possesses a few unique attributes including [JOBPROLOG](#), [JOBPILOG](#), [RESFAILPOLICY](#), and [DISABLEAM](#) attributes described in what follows:

### MANAGERS

Users listed via the **MANAGERS** parameter are granted full control over all jobs submitted to or running within the specified class.

```
# allow john and steve to cancel and modify all jobs submitted to the class/queue
special
CLASSCFG[special] MANAGERS=john,steve
```

In particular, a class manager can perform the following actions on jobs within a class/queue:

- view/diagnose job ([checkjob](#))
- cancel, requeue, suspend, resume, and checkpoint job ([mjobctl](#))
- modify job ([mjobctl](#))

### JOBPROLOG

The **JOBPROLOG** class performs a function similar to the resource manager level job prolog feature; however, there are some key differences:

- Moab prologs execute on the head node; resource manager prologs execute on the nodes allocated to the job.
- Moab prologs execute as the primary Moab administrator, resource manager prologs execute as root.
- Moab prologs can incorporate cluster environment information into their decisions and actions. (See [Valid Variables](#).)
- Unique Moab prologs can be specified on a per class basis.
- Job start requests are not sent to the resource manager until the Moab job prolog is successfully completed.
- Error messages generated by a Moab prolog are attached to jobs and associated objects; stderr from prolog script is attached to job.
- Moab prologs have access to Moab internal and peer services.

Valid epilog and prolog variables are:

| Variable      | Description                    |
|---------------|--------------------------------|
| <b>\$TIME</b> | Time that the trigger launches |
| <b>\$HOME</b> | Moab home directory            |

| Variable            | Description                        |
|---------------------|------------------------------------|
| <b>\$USER</b>       | User name the job is running under |
| <b>\$JOBID</b>      | Unique job identifier              |
| <b>\$HOSTLIST</b>   | Entire host list for job           |
| <b>\$MASTERHOST</b> | Master host for job                |

The **JOBPROLOG** class attribute allows a site to specify a unique per-class action to take before a job is allowed to start. This can be used for environmental provisioning, pre-execution resource checking, security management, and other functions. Sample uses may include enabling a VLAN, mounting a global file system, installing a new application or virtual node image, creating dynamic storage partitions, or activating job specific software services.

**i** A prolog is considered to have failed if it returns a negative number. If a prolog fails, the associated job will not start.

**i** If a prolog executes successfully, the associated epilog is guaranteed to start, even if the job fails for any reason. This allows the epilog to undo any changes made to the system by the prolog.

### Job Prolog Examples

```
# explicitly specify prolog arguments for special epilog
CLASSCFG[special] JOBPROLOG='$TOOLSDIR/specialprolog.pl $JOBID $HOSTLIST'
# use default prolog arguments for batch prolog
CLASSCFG[batch] JOBPROLOG=$TOOLSDIR/batchprolog.pl
```

### JOBEPILOG

The Moab epilog is nearly identical to the prolog in functionality except that it runs after the job completes within the resource manager but before the scheduler releases the allocated resources for use by subsequent jobs. It is commonly used for job clean-up, file transfers, signaling peer services, and undoing other forms of resource customization.

**i** An epilog is considered to have failed if it returns a negative number. If an epilog fails, the associated job will be annotated and a message will be sent to administrators.

### RESFAILPOLICY

This policy allows specification of the action to take on a per-class basis when a failure occurs on a node allocated to an actively running job. See the [Node Availability Overview](#) for more information.

## DISABLEAM

You can disable allocation management for jobs in specific classes by setting the **DISABLEAM** class attribute to **TRUE**. For all jobs outside of the specified classes, allocation enforcement will continue to be enforced.

```
# do not enforce allocations on low priority and debug jobs
CLASSCFG[lowprio]  DISABLEAM=TRUE
CLASSCFG[debug]    DISABLEAM=TRUE
```

## Setting Default Classes

In many cases, end-users do not want to be concerned with specifying a job class/queue. This is often handled by defining a default class. Whenever a user does not explicitly submit a job to a particular class, a default class, if specified, is used. In resource managers such as [TORQUE](#), this can be done at the resource manager level and its impact is transparent to the scheduler. The default class can also be enabled within the scheduler on a per resource manager or per user basis. To set a resource manager default class within Moab, use the **DEFAULTCLASS** attribute of the [RMCFG](#) parameter. For per user defaults, use the **CDEF** attribute of the [USERCFG](#) parameter.

## Creating a Remap Class

If a single default class is not adequate, Moab provides more flexible options with the [REMAPCLASS](#) parameter. If this parameter is set and a job is submitted to the remap class, Moab attempts to determine the final class to which a job belongs based on the resources requested. If a remap class is specified, Moab compares the job's requested nodes, processors, memory, and node features with the class's corresponding minimum and maximum resource limits. Classes are searched in the order in which they are defined; when the first match is found, Moab assigns the job to that class.

Because Moab remaps at job submission, updates you make to job requirements after submission will not cause any class changes. Moab does not restart the process.



In order to use **REMAPCLASS**, you must specify a **DEFAULTCLASS**. For example:

```
RMCFG[internal] DEFAULTCLASS=batch
```

In the example that follows, a job requesting 4 processors and the node feature *fast* are assigned to the class *quick*.

```
# You must specify a default class in order to use remap classes
RMCFG[internal]   DEFAULTCLASS=batch

# Jobs submitted to "batch" should be remapped
REMAPCLASS       batch

# stevens only queue
CLASSCFG[stevens] REQ.FEATURES=stevens REQUIREDUSERLIST=stevens,steven2

# Special queue for I/O nodes
CLASSCFG[io]      MAX.PROC=8 REQ.FEATURES=io

# General access queues
CLASSCFG[quick]   MIN.PROC=2 MAX.PROC=8 REQ.FEATURES=fast|short
CLASSCFG[medium]  MIN.PROC=2 MAX.PROC=8
CLASSCFG[DEFAULT] MAX.PROC=64
...
```

The following parameters can be used to remap jobs to different classes:

- **MIN.PROC**
- **MAX.PROC**
- **MIN.WCLIMIT**
- **MAX.WCLIMIT**
- **REQ.FEATURES**
- **REQ.FLAGS=INTERACTIVE**
- **REQUIREDUSERLIST**

If the parameter [REMAPCLASSLIST](#) is set, then only the listed classes are searched and they are searched in the order specified by this parameter. If none of the listed classes are valid for a particular job, that job retains its original class.

**i** The remap class only works with resource managers that allow dynamic modification of a job's assigned class/queue.

**i** If default credentials are specified on a remap class, a job submitted to that class will inherit those credentials. If the destination class has different default credentials, the new defaults override the original settings. If the destination class does not have default credentials, the job maintains the defaults inherited from the remap class.

## Class Attribute Overview

The following table enumerates the different parameters for **CLASSCFG**.

**i** Setting **DEFAULT.\*** on a class does not assign resources or features to that class. Rather, it specifies resources that jobs will inherit when they are submitted to the class without their own resource requests. To configure features, use [NODECFG](#).

**DEFAULT.ATTR**

|                    |                                                     |
|--------------------|-----------------------------------------------------|
| <b>Format</b>      | <ATTRIBUTE>[,<ATTRIBUTE>]...                        |
| <b>Description</b> | One or more comma-delimited generic job attributes. |
| <b>Example</b>     | ---                                                 |

**DEFAULT.DISK**

|                    |                                         |
|--------------------|-----------------------------------------|
| <b>Format</b>      | <INTEGER>                               |
| <b>Description</b> | Default amount of requested disk space. |
| <b>Example</b>     | ---                                     |

**DEFAULT.EXT**

|                    |                           |
|--------------------|---------------------------|
| <b>Format</b>      | <STRING>                  |
| <b>Description</b> | Default job RM extension. |
| <b>Example</b>     | ---                       |

**DEFAULT.FEATURESDEFAULT.EXT**

|                    |                                                                                                                               |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | Comma-delimited list of features.                                                                                             |
| <b>Description</b> | Default list of requested <a href="#">node features</a> (a.k.a, node properties). This only applies to compute resource reqs. |
| <b>Example</b>     | ---                                                                                                                           |

**DEFAULT.GRES**

|                    |                                                                                  |
|--------------------|----------------------------------------------------------------------------------|
| <b>Format</b>      | <STRING>[<COUNT>][,<STRING>[<COUNT>]]...                                         |
| <b>Description</b> | Default list of per task required <a href="#">consumable generic resources</a> . |

| DEFAULT.GRES   |                                               |
|----------------|-----------------------------------------------|
| <b>Example</b> | <code>CLASSCFG[viz] DEFAULT.GRES=viz:2</code> |

| DEFAULT.MEM        |                                     |
|--------------------|-------------------------------------|
| <b>Format</b>      | <INTEGER> (in MB)                   |
| <b>Description</b> | Default amount of requested memory. |
| <b>Example</b>     | ---                                 |

| DEFAULT.NODE       |                                                                                                                                                                                                                                                      |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <INTEGER>                                                                                                                                                                                                                                            |
| <b>Description</b> | Default required node count.                                                                                                                                                                                                                         |
| <b>Example</b>     | <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"><code>CLASSCFG[viz] DEFAULT.NODE=5</code></div> <p>When a user submits a job to the <code>viz</code> class without a specified node count, the job is assigned 5 nodes.</p> |

| DEFAULT.NODESET    |                                                                                                                                        |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <SETTYPE>:<SETATTR>[:<SETLIST>[,<SETLIST>]...]                                                                                         |
| <b>Description</b> | Default <a href="#">node set</a> .                                                                                                     |
| <b>Example</b>     | <div style="border: 1px dashed gray; padding: 5px;"><code>CLASSCFG [amd]<br/>DEFAULT.NODESET=ONEOF:FEATURE:ATHLON,OPTERON</code></div> |

| DEFAULT.PROC       |                                         |
|--------------------|-----------------------------------------|
| <b>Format</b>      | <INTEGER>                               |
| <b>Description</b> | Default number of requested processors. |
| <b>Example</b>     | ---                                     |

**DEFAULT.TPN**

|                    |                                   |
|--------------------|-----------------------------------|
| <b>Format</b>      | <INTEGER>                         |
| <b>Description</b> | Default number of tasks per node. |
| <b>Example</b>     | ---                               |

**DEFAULT.WCLIMIT**

|                    |                          |
|--------------------|--------------------------|
| <b>Format</b>      | <INTEGER>                |
| <b>Description</b> | Default wallclock limit. |
| <b>Example</b>     | ---                      |

**EXCL.FEATURES**

|                    |                                                                                                                                                                                                |
|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | Comma- or pipe-delimited list of node features.                                                                                                                                                |
| <b>Description</b> | Set of excluded (disallowed) features. If delimited by commas, reject job if all features are requested; if delimited by the pipe symbol ( ), reject job if at least one feature is requested. |
| <b>Example</b>     | <code>CLASSCFG[intel] EXCL.FEATURES=ATHLON,AMD</code>                                                                                                                                          |

**EXCL.FLAGS**

|                    |                                                                                 |
|--------------------|---------------------------------------------------------------------------------|
| <b>Format</b>      | Comma-delimited list of <a href="#">job flags</a> .                             |
| <b>Description</b> | Set of excluded (disallowed) job flags. Reject job if any listed flags are set. |
| <b>Example</b>     | <code>CLASSCFG[batch] EXCL.FLAGS=INTERACTIVE</code>                             |

**EXCLUDEUSERLIST**

|               |                                |
|---------------|--------------------------------|
| <b>Format</b> | Comma-delimited list of users. |
|---------------|--------------------------------|

| EXCLUDEUSERLIST    |                                              |
|--------------------|----------------------------------------------|
| <b>Description</b> | List of users not permitted access to class. |
| <b>Example</b>     | ---                                          |

| FORCENODEACCESSPOLICY |                                                                                                                                                                                                             |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>         | one of <i>SINGLETASK</i> , <i>SINGLEJOB</i> , <i>SINGLEUSER</i> , or <i>SHARED</i>                                                                                                                          |
| <b>Description</b>    | Node access policy associated with queue. If set, this value overrides any per job settings specified by the user at the job level. (See <a href="#">Node Access Policy</a> overview for more information.) |
| <b>Example</b>        | <code>CLASSCFG [batch] FORCENODEACCESSPOLICY=SINGLEJOB</code>                                                                                                                                               |

| FSCAP              |                                                       |
|--------------------|-------------------------------------------------------|
| <b>Format</b>      | <DOUBLE>[%]                                           |
| <b>Description</b> | See <a href="#">fairshare policies</a> specification. |
| <b>Example</b>     | ---                                                   |

| FSTARGET           |                                                       |
|--------------------|-------------------------------------------------------|
| <b>Format</b>      | <DOUBLE>[%]                                           |
| <b>Description</b> | See <a href="#">fairshare policies</a> specification. |
| <b>Example</b>     | ---                                                   |

| HOSTLIST           |                                                                                                                                               |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <a href="#">Host expression</a> , or comma-delimited list of hosts or host ranges.                                                            |
| <b>Description</b> | List of hosts associated with a class. If specified, Moab constrains the availability of a class to only nodes listed in the class host list. |

## HOSTLIST

|                |                                                     |
|----------------|-----------------------------------------------------|
| <b>Example</b> | <code>CLASSCFG[batch] HOSTLIST=r:abs[45-113]</code> |
|----------------|-----------------------------------------------------|

## JOBPILOG

|                    |                                                                                                                                   |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <STRING>                                                                                                                          |
| <b>Description</b> | Scheduler level job epilog to be run after job is completed by resource manager. (See <a href="#">special class attributes</a> .) |
| <b>Example</b>     | ---                                                                                                                               |

## JOBFLAGS

|                    |                                                                               |
|--------------------|-------------------------------------------------------------------------------|
| <b>Format</b>      | Comma-delimited list of job flags.                                            |
| <b>Description</b> | See the <a href="#">flag overview</a> for a description of legal flag values. |
| <b>Example</b>     | <code>CLASSCFG[batch] JOBFLAGS=restartable</code>                             |

## JOBPROLOG

|                    |                                                                                                                                  |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <STRING>                                                                                                                         |
| <b>Description</b> | Scheduler level job prolog to be run before job is started by resource manager. (See <a href="#">special class attributes</a> .) |
| <b>Example</b>     | ---                                                                                                                              |

## MANAGERS

|                    |                                                                                                                                 |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <USER>[,<USER>]..                                                                                                               |
| <b>Description</b> | Users allowed to control, cancel, preempt, and modify jobs within class/queue. (See <a href="#">special class attributes</a> .) |
| <b>Example</b>     | <code>CLASSCFG[fast] MANAGERS=root,kerry,e43</code>                                                                             |

| MAXJOB             |                                              |
|--------------------|----------------------------------------------|
| <b>Format</b>      | <INTEGER>                                    |
| <b>Description</b> | Maximum number of jobs allowed in the class. |
| <b>Example</b>     | ---                                          |

| MAXPROCPERNODE           |                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Form-<br/>at</b>      | <INTEGER>                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>Descri-<br/>ption</b> | Maximum number of processors requested per node. May optionally include node names to articulate which nodes have a specific limit.                                                                                                                                                                                                                                                             |
| <b>Exam-<br/>ple</b>     | <pre>CLASSCFG[cpu] MAXPROCPERNODE=20 # When using this class, limit 20 for all nodes</pre> <pre>CLASSCFG[cpu] MAXPROCPERNODE[n1,n2]=20 MAXPROCPERNODE[n3]=10 # When using this class, limit 20 for n1 &amp; n2 and limit 10 for n3</pre> <pre>CLASSCFG[cpu] MAXPROCPERNODE[n1,n2]=20 MAXPROCPERNODE=10 # When using this class, limit 20 for n1 &amp; n2 and limit 10 for all other nodes</pre> |

| MAX.CPUTIME        |                                    |
|--------------------|------------------------------------|
| <b>Format</b>      | <INTEGER>                          |
| <b>Description</b> | Maximum allowed utilized CPU time. |
| <b>Example</b>     | ---                                |

| MAX.NODE           |                                                                                                                           |
|--------------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <INTEGER>                                                                                                                 |
| <b>Description</b> | Maximum number of requested nodes per job. (Also used when <a href="#">REMAPCLASS</a> is set to correctly route the job.) |

## MAX.NODE

## Example

```
CLASSCFG[batch] MAX.NODE=64
```

Deny jobs requesting over 64 nodes access to the class *batch*.

## MAX.PROC

## Format

<INTEGER>

## Description

Maximum number of requested processors per job. (Also used when [REMAPCLASS](#) is set to correctly route the job.)

## Example

```
CLASSCFG[small] MAX.PROC[USER]=3,6
```

## MAX.PS

## Format

<INTEGER>

## Description

Maximum requested processor-seconds.

## Example

---

## MAX.WCLIMIT

## Format

[[[DD:]HH:]MM:]SS

## Description

Maximum allowed wallclock limit per job. (Also used when [REMAPCLASS](#) is set to correctly route the job.)

## Example

```
CLASSCFG[long] MAX.WCLIMIT=96:00:00
```

## MIN.NODE

## Format

<INTEGER>

## Description

Minimum number of requested nodes per job. (Also used when [REMAPCLASS](#) is set to correctly route the job.)

| MIN.NODE       |                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------|
| <b>Example</b> | <pre>CLASSCFG[dev] MIN.NODE=16</pre> <p>Jobs must request at least 16 nodes to be allowed to access the class.</p> |

| MIN.PROC           |                                                                                                                                |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <INTEGER>                                                                                                                      |
| <b>Description</b> | Minimum number of requested processors per job. (Also used when <a href="#">REMAPCLASS</a> is set to correctly route the job.) |
| <b>Example</b>     | <pre>CLASSCFG[dev] MIN.PROC=32</pre> <p>Jobs must request at least 32 processors to be allowed to access the class.</p>        |

| MIN.PS             |                                      |
|--------------------|--------------------------------------|
| <b>Format</b>      | <INTEGER>                            |
| <b>Description</b> | Minimum requested processor-seconds. |
| <b>Example</b>     | ---                                  |

| MIN.TPN            |                                          |
|--------------------|------------------------------------------|
| <b>Format</b>      | <INTEGER>                                |
| <b>Description</b> | Minimum required tasks per node per job. |
| <b>Example</b>     | ---                                      |

| MIN.WCLIMIT        |                                                                                                                          |
|--------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | [[[DD:]HH:]MM:]SS                                                                                                        |
| <b>Description</b> | Minimum required wallclock limit per job. (Also used when <a href="#">REMAPCLASS</a> is set to correctly route the job.) |

## MIN.WCLIMIT

**Example** ---

## NODEACCESSPOLICY

**Format** one of *SINGLETASK*, *SINGLEJOB*, *SINGLEUSER*, or *SHARED*

**Description** Default node access policy associated with queue. This value will be overridden by any per job settings specified by the user at the job level. (See [Node Access Policy](#) overview.)

**Example** `CLASSCFG [batch] NODEACCESSPOLICY=SINGLEJOB`

## PARTITION

**Format** <STRING>

**Description** Partition name where jobs associated with this class must run.

**Example** `CLASSCFG [batch] PARTITION=p12`

## PRIORITY

**Format** <INTEGER>

**Description** Priority associated with the class. (See [Priority](#) overview.)

**Example** `CLASSCFG [batch] PRIORITY=1000`

## QDEF

**Format** <QOSID>

**Description** Default QoS for jobs submitted to this class. You may specify a maximum of four **QDEF** entries per credential. Any QoS specified after the fourth will not be accepted.

 In addition to classes, you may also specify **QDEF** for accounts, groups, and users.

| QDEF           |                                                                                                                                                               |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Example</b> | <pre>CLASSCFG[batch] QDEF=base</pre> <p>Jobs submitted to class <i>batch</i> that do not explicitly request a QoS will have the QoS <i>base</i> assigned.</p> |

| QLIST              |                                                                    |
|--------------------|--------------------------------------------------------------------|
| <b>Format</b>      | <QOSID>[,<QOSID>]..                                                |
| <b>Description</b> | List of accessible QoSs for jobs submitted to this class.          |
| <b>Example</b>     | <pre>CLASSCFG[batch] QDEF=base QLIST=base,fast,special,bigio</pre> |

| REQ.FEATURES       |                                                                                                                                                     |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | Comma- or pipe-delimited list of node features.                                                                                                     |
| <b>Description</b> | Set of required features. If delimited by commas, all features are required; if delimited by the pipe symbol ( ), at least one feature is required. |
| <b>Example</b>     | <pre>CLASSCFG[amd] REQ.FEATURES=ATHLON,AMD</pre>                                                                                                    |

| REQ.FLAGS          |                                                       |
|--------------------|-------------------------------------------------------|
| <b>Format</b>      | REQ.FLAGS can be used with only the INTERACTIVE flag. |
| <b>Description</b> | Sets the INTERACTIVE flag on jobs in this class.      |
| <b>Example</b>     | <pre>CLASSCFG[orion] REQ.FLAGS=INTERACTIVE</pre>      |

| REQUIREDACCOUNTLIST |                                                                                                       |
|---------------------|-------------------------------------------------------------------------------------------------------|
| <b>Format</b>       | Comma-delimited list of accounts.                                                                     |
| <b>Description</b>  | List of accounts allowed to access and use a class (analogous to <i>*LIST</i> for other credentials). |

### REQUIREDACCOUNTLIST

**Example**

```
CLASSCFG[jasper] REQUIREDACCOUNTLIST=testers,development
```

### REQUIREDUSERLIST

**Format**

Comma-delimited list of users.

**Description**

List of users allowed to access and use a class (analogous to **\*LIST** for other credentials).

**Example**

```
CLASSCFG[jasper] REQUIREDUSERLIST=john,u13,steve,guest
```

### REQUIREDQOSLIST

**Format**

Comma-delimited list of QoSs

**Description**

List of QoSs allowed to access and use a class (analogous to **\*LIST** for other credentials).



The number of unique QoSs is limited by the Moab Maximum ACL limit, which defaults to 32.

**Example**

```
CLASSCFG[jasper] REQUIREDQOSLIST=hi,lo
```

### SYSPRIO

**Format**

<INTEGER>

**Description**

Value of [system priority](#) applied to every job submitted to this class.

**Example**

```
CLASSCFG[special] SYSPRIO=100
```

### WCOVERRUN

**Format**

[[[DD:]HH:]MM:]SS

| WCOVERRUN          |                                                                |
|--------------------|----------------------------------------------------------------|
| <b>Description</b> | Tolerated amount of time beyond the specified wallclock limit. |
| <b>Example</b>     | ---                                                            |

## Enabling Queue Complex Functionality

Queue complexes allow an organization to build a hierarchy of queues and apply certain limits and rules to collections of these queues. Moab supports this functionality in two ways. The first way, queue mapping, is very simple but limited in functionality. The second method provides very rich functionality but requires more extensive configuration using the Moab hierarchical fairshare facility.

### Queue Mapping

Queue mapping allows collections of queues to be mapped to a parent credential object against which various limits and policies can be applied, as in the following example.

```

QOSCFG[general]   MAXIJOB[USER]=14  PRIORITY=20
QOSCFG[prio]     MAXIJOB[USER]=8   PRIORITY=2000
# group short, med, and long jobs into 'general' QOS
CLASSCFG[short]  QDEF=general FSTARGET=30
CLASSCFG[med]    QDEF=general FSTARGET=40
CLASSCFG[long]   QDEF=general FSTARGET=30 MAXPROC=200
# group interactive and debug jobs into 'prio' QOS
CLASSCFG[inter]  QDEF=prio
CLASSCFG[debug]  QDEF=prio
CLASSCFG[premier] PRIORITY=10000

```

## QoS Credential

The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoSs differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.

QoS access and QoS defaults can be mapped to users, groups, accounts, and classes, allowing limited service offering for key users. As mentioned, these services focus around increasing access to special scheduling capabilities & additional resources and improving job responsiveness. At a high level, unique QoS attributes can be broken down into the following:

- [Usage Limit Overrides](#)
- [Service Targets](#)
- [Privilege Flags](#)

- [Charge Rate](#)
- [Access Controls](#)

## QoS Usage Limit Overrides

All credentials allow specification of job limits. In such cases, jobs are constrained by the most limiting of all applicable policies. With QoS override limits, however, jobs are limited by the override, regardless of other limits specified.

## QoS Service Targets

Service targets cause the scheduler to take certain job-related actions as various responsiveness targets are met. Targets can be set for either job queue time or job expansion factor and cause priority adjustments, reservation enforcement, or preemption activation. In strict service centric organizations, Moab can be configured to trigger various events and notifications in the case of failure by the cluster to meet responsiveness targets.

## QoS Privilege Flags

QoSs can provide access to special capabilities. These capabilities include preemption, job deadline support, backfill, next to run priority, guaranteed resource reservation, resource provisioning, dedicated resource access, and many others. See the complete list in the [QoS Facility Overview](#) section.

## QoS Charge Rate

Associated with the QoSs many privileges is the ability to assign end-users costs for the use of these services. This charging can be done on a per-QoS basis and may be specified for both dedicated and use-based resource consumption. The [Per QoS Charging](#) section covers more details on QoS level costing configuration while the Charging and Allocation Management section provides more details regarding general single cluster and multi-cluster charging capabilities.

## QoS Access Controls

QoS access control can be enabled on a per QoS basis using the [MEMBERULIST](#) attribute or specified on a *per-requestor* basis using the [QDEF](#) and [QLIST](#) attributes of the [USERCFG](#), [GROUPCFG](#), [ACCOUNTCFG](#), and [CLASSCFG](#) parameters. See [Managing QoS Access](#) for more detail.

### Related topics

- [Identity Manager Interface](#)
- [Usage Limits](#)

# Job Attributes/Flags Overview

## Job Attributes

| FLAGS               |                                                                                                                                                                                                   |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | <FLAG>[:<FLAG>]...                                                                                                                                                                                |
| <b>Default:</b>     | ---                                                                                                                                                                                               |
| <b>Description:</b> | Specifies job specific flags.                                                                                                                                                                     |
| <b>Example:</b>     | <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 5px;"> <pre>FLAGS=ADVRES:RESTARTABLE</pre> </div> <p><i>The job can restart and should only utilize reserved resources.</i></p> |

| PLIST*              |                                                                                                                                                                                                                                               |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | <PARTITION_NAME>[^ &]<br>[:<PARTITION_NAME>[^ &]]...                                                                                                                                                                                          |
| <b>Default:</b>     | <i>[ALL]</i>                                                                                                                                                                                                                                  |
| <b>Description:</b> | Specifies the list of partitions the object can access. If no partition list is specified, the object is granted default access to all partitions.                                                                                            |
| <b>Example:</b>     | <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 5px;"> <pre>PLIST=OldSP:Cluster1:O3K</pre> </div> <p><i>The object can access resources located in the <i>OldSP</i>, <i>Cluster1</i>, and/or <i>O3K</i> partitions.</i></p> |

| QDEF                |                                                       |
|---------------------|-------------------------------------------------------|
| <b>Format:</b>      | <QOS_NAME>                                            |
| <b>Default:</b>     | <i>[DEFAULT]</i>                                      |
| <b>Description:</b> | Specifies the default QOS associated with the object. |

| QDEF            |                                                                                       |
|-----------------|---------------------------------------------------------------------------------------|
| <b>Example:</b> | <pre>QDEF=premium</pre> <p><i>The object is assigned the default QoS premium.</i></p> |

| QLIST*              |                                                                                                                                            |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | <QOS_NAME>[^ &]<br>[:<QOS_NAME>[^ &]]...                                                                                                   |
| <b>Default:</b>     | <QDEF>                                                                                                                                     |
| <b>Description:</b> | Specifies the list of QoSs the object can access. If no QoS list is specified, the object is granted access only to its default partition. |
| <b>Example:</b>     | <pre>QLIST=premium:express:bottomfeeder</pre> <p><i>The object can access any of the 3 QoSs listed.</i></p>                                |

**i** By default, jobs may access QoSs based on the 'logical or' of the access lists associated with all job credentials. For example, a job associated with user "John," group "staff," and class "batch" may utilize QoSs accessible by any of the individual credentials. Thus the job's QoS access list, or QLIST, equals the 'or' of the user, group, and class QLIST's. (i.e., JOBQLIST = USERQLIST | GROUPQLIST | CLASSQLIST). If the ampersand symbol, '&', is associated with any list, this list is logically and'd with the other lists. If the carat symbol, '^', is associated with any object QLIST, this list is exclusively set, regardless of other object access lists using the following order of precedence user, group, account, QOS, and class. These special symbols affect the behavior of both QOS and partition access lists.

## Job Flags

| ADVRES              |                                                                                                                                                          |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | ADVRES[:<RESID>]                                                                                                                                         |
| <b>Default:</b>     | Use available resources where ever found, whether inside a reservation or not.                                                                           |
| <b>Description:</b> | Specifies the job may only utilize accessible, reserved resources. If <RESID> is specified, only resources in the specified reservation may be utilized. |

## ADVRES

**Example:**

```
FLAGS=ADVRES:MET.A.1
```

*The job may only utilize resources located in the **META.1** reservation.*

## ARRAYJOBPARLOCK

**Format:**

---

**Default:**

---

**Description:**

Specifies that the job array being submitted should not span across multiple partitions. This locks all sub jobs of the array to a single partition. If you want to lock all job arrays to a single partition, specify the [ARRAYJOBPARLOCK](#) parameter in `moab.cfg` to force this behavior on a global scale.

**Example:**

```
> msub -t moab.[1-5]%3 -l walltime=30,flags=arrayjobparlock
```

## ARRAYJOBPARSPAN

**Format:**

---

**Default:**

---

**Description:**

Specifies that the job array being submitted should span across multiple partitions. This is the default behavior in Moab, unless the [ARRAYJOBPARLOCK](#) parameter is specified in `moab.cfg`. This job flag overrides the **ARRAYJOBPARLOCK** parameter so that job arrays can be allowed to span multiple partitions at submit time.

**Example:**

```
> msub -t moab.[1-5]%3 -l walltime=30,flags=arrayjobparspan
```

## GRESONLY

**Format:**

GRESONLY

**Default:**

False

**Description:**

Uses no compute resources such as processors, memory, and so forth; uses only generic resources.

## GRESONLY

**Example:**

```
> msub -l gres=matlab,walltime=300
```

## IGNIDLEJOBRSV

**Format:**

IGNIDLEJOBRSV

**Default:**

N/A

**Description:**

Only applies to QOS. **IGNIDLEJOBRSV** allows jobs to start without a guaranteed walltime. Instead, it overlaps the idle reservations of real jobs and is preempted 2 minutes before the real job starts.

**Example:**

```
QOSCFG[standby] JOBFLAGS=IGNIDLEJOBRSV
```

## NOQUEUE

**Format:**

NOQUEUE

**Default:**

Jobs remain queued until they are able to run

**Description:**

Specifies that the job should be removed if it is unable to allocate resources and start execution immediately.

**Example:**

```
FLAGS=NOQUEUE
```

*The job should be removed unless it can start running at submit time.*

This functionality is identical to the resource manager extension [QUEUEJOB:FALSE](#).

## NORMSTART

**Format:**

NORMSTART

**Default:**

Moab passes jobs to a resource manager to schedule.

**Description:**

Specifies that the job is an internal system job and will not be started via an RM.

| NORMSTART       |                                                                                                         |
|-----------------|---------------------------------------------------------------------------------------------------------|
| <b>Example:</b> | <pre>FLAGS=NORMSTART</pre> <p><i>The job begins running in Moab without a corresponding RM job.</i></p> |

| NOVMMIGRATE        |                                                                                                                                                 |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | NOVMMIGRATE                                                                                                                                     |
| <b>Default</b>     | Moab can migrate the VM associated with the job.                                                                                                |
| <b>Description</b> | Specifies that Moab may not migrate the VM that the job sets up.                                                                                |
| <b>Example</b>     | <pre>msub -l walltime=INFINITY,template=VMTracking,os=linux,nodes=h3,jobflags=novmmigrate</pre> <p><i>Moab will not migrate the new VM.</i></p> |

| PREEMPTEE           |                                                                                                                           |
|---------------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | PREEMPTEE                                                                                                                 |
| <b>Default:</b>     | Jobs may not be preempted by other jobs                                                                                   |
| <b>Description:</b> | Specifies that the job may be <a href="#">preempted</a> by other jobs which have the <b>PREEMPTOR</b> flag set.           |
| <b>Example:</b>     | <pre>FLAGS=PREEMPTEE</pre> <p><i>The job may be preempted by other jobs which have the <b>PREEMPTOR</b> flag set.</i></p> |

| PREEMPTOR           |                                                                                                          |
|---------------------|----------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | PREEMPTOR                                                                                                |
| <b>Default:</b>     | Jobs may not preempt other jobs                                                                          |
| <b>Description:</b> | Specifies that the job may <a href="#">preempt</a> other jobs which have the <b>PREEMPTEE</b> flag set . |

| PREEMPTOR       |                                                                                                                   |
|-----------------|-------------------------------------------------------------------------------------------------------------------|
| <b>Example:</b> | <pre>FLAGS=PREEMPTOR</pre> <p><i>The job may preempt other jobs which have the <b>PREEMPTEE</b> flag set.</i></p> |

| PURGEONSUCCESSONLY |                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | PURGEONSUCCESSONLY                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Default</b>     | Completed jobs are sent to a queue for a short period of time before Moab purges them from the system.                                                                                                                                                                                                                                                                                  |
| <b>Description</b> | Specifies that Moab should only purge the job from the completed queue if it completed successfully. If the job failed, Moab will keep it in the queue indefinitely to allow you to restart it at any time. This flag is particularly useful for setup and take down jobs in job workflows. See <a href="#">Creating workflows with job templates on page 857</a> for more information. |
| <b>Example</b>     | <pre>FLAGS=PURGEONSUCCESSONLY</pre> <p><i>If the job fails, Moab will not purge it from the completed job queue.</i></p>                                                                                                                                                                                                                                                                |

| RESTARTABLE         |                                                                                                               |
|---------------------|---------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | RESTARTABLE                                                                                                   |
| <b>Default:</b>     | Jobs may not be restarted if preempted.                                                                       |
| <b>Description:</b> | Specifies jobs can be <i>requeued</i> and later restarted if <a href="#">preempted</a> .                      |
| <b>Example:</b>     | <pre>FLAGS=RESTARTABLE</pre> <p><i>The associated job can be preempted and restarted at a later date.</i></p> |

| SUSPENDABLE    |             |
|----------------|-------------|
| <b>Format:</b> | SUSPENDABLE |

| SUSPENDABLE         |                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------|
| <b>Default:</b>     | Jobs may not be suspended if preempted.                                                                     |
| <b>Description:</b> | Specifies jobs can be <i>suspended</i> and later resumed if <a href="#">preempted</a> .                     |
| <b>Example:</b>     | <pre>FLAGS=SUSPENDABLE</pre> <p><i>The associated job can be suspended and resumed at a later date.</i></p> |

| SYSTEMJOB           |                                                                 |
|---------------------|-----------------------------------------------------------------|
| <b>Format:</b>      | SYSTEMJOB                                                       |
| <b>Default:</b>     | N/A                                                             |
| <b>Description:</b> | Creates an internal system job that does not require resources. |
| <b>Example:</b>     | <pre>FLAGS=SYSTEMJOB</pre>                                      |

| WIDERSVSEARCHALGO   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format:</b>      | <BOOLEAN>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Default:</b>     | ---                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>Description:</b> | When Moab is determining when and where a job can run, it either searches for the most resources or the longest range of resources. In almost all cases searching for the longest range is ideal and returns the soonest starttime. In some rare cases, however, a particular job may need to search for the most resources. In those cases this flag can be used to have the job find the soonest starttime. The flag can be specified at submit time, or you can use <a href="#">mjobctl -m</a> to modify the job after it has been submitted. See the <a href="#">RSVSEARCHALGO</a> parameter. |
| <b>Example:</b>     | <pre>&gt; msub -l flags=widersvsearchalgo &gt; mjobctl -m flags+=widersvsearchalgo job.1</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

### Related topics

- [Setting Per-Credential Job Flags](#)

# Scheduler Commands

## Moab Commands

| Command                                 | Description                                                          |
|-----------------------------------------|----------------------------------------------------------------------|
| <a href="#"><u>checkjob</u></a>         | Provide detailed status report for specified job                     |
| <a href="#"><u>checknode</u></a>        | Provide detailed status report for specified node                    |
| <a href="#"><u>mcredctl</u></a>         | Controls various aspects about the credential objects within Moab    |
| <a href="#"><u>mdiag</u></a>            | Provide diagnostic reports for resources, workload, and scheduling   |
| <a href="#"><u>mjobctl</u></a>          | Control and modify job                                               |
| <a href="#"><u>mnodectl</u></a>         | Control and modify nodes                                             |
| <a href="#"><u>moab</u></a>             | Control the Moab daemon                                              |
| <a href="#"><u>mrmctl</u></a>           | Query and control resource managers                                  |
| <a href="#"><u>mrsvctl</u></a>          | Create, control and modify reservations                              |
| <a href="#"><u>mschedctl</u></a>        | Modify scheduler state and behavior                                  |
| <a href="#"><u>mshow</u></a>            | Displays various diagnostic messages about the system and job queues |
| <a href="#"><u>mshow -a</u></a>         | Query and show available system resources                            |
| <a href="#"><u>msub</u></a>             | Scheduler job submission                                             |
| <a href="#"><u>mvcctl</u></a>           | Create, modify, and delete VCs                                       |
| <a href="#"><u>mvmctl</u></a>           | Create, control and modify VMs                                       |
| <a href="#"><u>showbf</u></a>           | Show current resource availability                                   |
| <a href="#"><u>showhist.moab.pl</u></a> | Show past job information                                            |

| Command                             | Description                                          |
|-------------------------------------|------------------------------------------------------|
| <a href="#"><u>showq</u></a>        | Show queued jobs                                     |
| <a href="#"><u>showres</u></a>      | Show existing reservations                           |
| <a href="#"><u>showstart</u></a>    | Show estimates of when job can/will start            |
| <a href="#"><u>showstate</u></a>    | Show current state of resources                      |
| <a href="#"><u>showstats</u></a>    | Show usage statistics                                |
| <a href="#"><u>showstats -f</u></a> | Show various tables of scheduling/system performance |

## Moab command options

For many Moab commands, you can use the following options to specify that Moab will run the command in a different way or different location from the configured default. These options do not change your settings in the configuration file; they override the settings for this single instance of the command.

| Option                                | Description                                                                                                                                               |
|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>--about</b>                        | Displays build and version information and the status of your Moab license                                                                                |
| <b>--help</b>                         | Displays usage information about the command                                                                                                              |
| <b>--host=&lt;server-HostName&gt;</b> | Causes Moab to run the client command on the specified host                                                                                               |
| <b>--loglevel=&lt;logLevel&gt;</b>    | Causes Moab to write log information to STDERR as the client command is running. For more information, see <a href="#">Logging Overview on page 676</a> . |
| <b>--msg=&lt;message&gt;</b>          | Causes Moab to annotate the action in the <a href="#">event log</a>                                                                                       |
| <b>--port=&lt;server-Port&gt;</b>     | Causes Moab to run the command using the port specified                                                                                                   |
| <b>--timeout=&lt;seconds&gt;</b>      | Sets the maximum time that the client command will wait for a response from the Moab server                                                               |

| Option           | Description                                            |
|------------------|--------------------------------------------------------|
| <b>--version</b> | Displays version information                           |
| <b>--xml</b>     | Causes Moab to return the command output in XML format |

## Commands Providing Maui Compatibility



The following commands are deprecated. Click the link for respective deprecated commands to see the updated replacement command for each.

| Command                            | Description                                                                          |
|------------------------------------|--------------------------------------------------------------------------------------|
| <a href="#"><u>canceljob</u></a>   | Cancel job                                                                           |
| <a href="#"><u>changeparam</u></a> | Change in memory parameter settings                                                  |
| <a href="#"><u>diagnose</u></a>    | Provide diagnostic report for various aspects of resources, workload, and scheduling |
| <a href="#"><u>releasehold</u></a> | Release job defers and holds                                                         |
| <a href="#"><u>releaseres</u></a>  | Release reservations                                                                 |
| <a href="#"><u>runjob</u></a>      | Force a job to run immediately                                                       |
| <a href="#"><u>sethold</u></a>     | Set job holds                                                                        |
| <a href="#"><u>setqos</u></a>      | Modify job QOS settings                                                              |
| <a href="#"><u>setres</u></a>      | Set an admin/user reservation                                                        |
| <a href="#"><u>setspri</u></a>     | Adjust job/system priority of job                                                    |
| <a href="#"><u>showconfig</u></a>  | Show current scheduler configuration                                                 |

## Status Commands

The status commands organize and present information about the current state and historical statistics of the scheduler, jobs, resources, users, and accounts. The following table presents the primary status commands and flags.

| Command                                                         | Description                                                                                                                                                                |
|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#"><u>checkjob</u></a>                                 | Displays detailed job information such as job state, resource requirements, environment, constraints, credentials, history, allocated resources, and resource utilization. |
| <a href="#"><u>checknode</u></a>                                | Displays detailed node information such as node state, resources, attributes, reservations, history, and statistics.                                                       |
| <a href="#"><u>mdiag -f</u></a>                                 | Displays summarized fairshare information and any unexpected fairshare configuration.                                                                                      |
| <a href="#"><u>mdiag -j</u></a>                                 | Displays summarized job information and any unexpected job state.                                                                                                          |
| <a href="#"><u>mdiag -n</u></a>                                 | Displays summarized node information and any unexpected node state.                                                                                                        |
| <a href="#"><u>mdiag -p</u></a>                                 | Displays summarized job priority information.                                                                                                                              |
| <a href="#"><u>mschedctl</u></a><br><a href="#"><u>-f</u></a>   | Resets internal statistics.                                                                                                                                                |
| <a href="#"><u>showstats</u></a><br><a href="#"><u>-f</u></a>   | Displays various aspects of scheduling performance across a job duration/job size matrix.                                                                                  |
| <a href="#"><u>showq [-r]-</u></a><br><a href="#"><u>i]</u></a> | Displays various views of currently queued active, idle, and non-eligible jobs.                                                                                            |
| <a href="#"><u>showstats</u></a><br><a href="#"><u>-g</u></a>   | Displays current and historical usage on a per group basis.                                                                                                                |
| <a href="#"><u>showstats</u></a><br><a href="#"><u>-u</u></a>   | Displays current and historical usage on a per user basis.                                                                                                                 |
| <a href="#"><u>showstats</u></a><br><a href="#"><u>-v</u></a>   | Displays high level current and historical scheduling statistics.                                                                                                          |

## Job Management Commands

Moab shares job management tasks with the resource manager. Typically, the scheduler only modifies scheduling relevant aspects of the job such as partition access, job priority, charge account, and hold state. The following table covers the available job management commands. The [Commands Overview](#) lists all available commands.

| Command                               | Description                                                                                                                               |
|---------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#"><u>canceljob</u></a>      | Cancels existing job.                                                                                                                     |
| <a href="#"><u>checkjob</u></a>       | Displays job state, resource requirements, environment, constraints, credentials, history, allocated resources, and resource utilization. |
| <a href="#"><u>mdiag -j</u></a>       | Displays summarized job information and any unexpected job state.                                                                         |
| <a href="#"><u>releasehold -a</u></a> | Removes job holds or deferrals.                                                                                                           |
| <a href="#"><u>runjob</u></a>         | Starts job immediately, if possible.                                                                                                      |
| <a href="#"><u>sethold</u></a>        | Sets hold on job.                                                                                                                         |
| <a href="#"><u>setqos</u></a>         | Sets/modifies QoS of existing job.                                                                                                        |
| <a href="#"><u>setspri</u></a>        | Adjusts job/system priority of job.                                                                                                       |

#### Related topics

- [Job State Definitions](#)

## Reservation Management Commands

Moab exclusively controls and manages all advance reservation features including both standing and administrative reservations. The following table covers the available reservation management commands.

| Command                           | Description                                                           |
|-----------------------------------|-----------------------------------------------------------------------|
| <a href="#"><u>mdiag -r</u></a>   | Displays summarized reservation information and any unexpected state. |
| <a href="#"><u>mrsvctl</u></a>    | Reservation control.                                                  |
| <a href="#"><u>mrsvctl -r</u></a> | Removes reservations.                                                 |
| <a href="#"><u>mrsvctl -c</u></a> | Creates an administrative reservation.                                |
| <a href="#"><u>showres</u></a>    | Displays information regarding location and state of reservations.    |

## Policy/Configuration Management Commands

Moab allows dynamic modification of most scheduling parameters allowing new scheduling policies, algorithms, constraints, and permissions to be set at any time. Changes made via Moab client commands are temporary and are overridden by values specified in Moab configuration files the next time Moab is shut down and restarted. The following table covers the available configuration management commands.

| Command                             | Description                                                                |
|-------------------------------------|----------------------------------------------------------------------------|
| <a href="#"><u>mschedctl -l</u></a> | Displays triggers, messages, and settings of all configuration parameters. |
| <a href="#"><u>mschedctl</u></a>    | Controls the scheduler (behavior, parameters, triggers, messages).         |
| <a href="#"><u>mschedctl -m</u></a> | Modifies system values.                                                    |

## End-user Commands

While the majority of Moab commands are tailored for use by system administrators, a number of commands are designed to extend the knowledge and capabilities of end-users. The following table covers the commands available to end-users.

**i** When using Active Directory as a central authentication mechanism, all nodes must be reported with a different name when booted in both Linux and Windows (for instance, `node01-1` for Linux and `node01` for Windows). If a machine account with the same name is created for each OS, the most recent OS will remove the previously-joined machine account. The nodes must report to Moab with the same hostname. This can be done by using aliases (adding all node names to the `/etc/hosts` file on the system where Moab is running) and ensuring that the Linux resource manager reports the node with its global name rather than the Linux-specific one (`node01` rather than `node01-1`).

| Command                           | Description                                                                                                                               |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#"><u>canceljob</u></a>  | Cancels existing job.                                                                                                                     |
| <a href="#"><u>checkjob</u></a>   | Displays job state, resource requirements, environment, constraints, credentials, history, allocated resources, and resource utilization. |
| <a href="#"><u>msub</u></a>       | Submit a new job.                                                                                                                         |
| <a href="#"><u>releaseres</u></a> | Releases a <a href="#"><u>user reservation</u></a> .                                                                                      |

| Command                          | Description                                                                                        |
|----------------------------------|----------------------------------------------------------------------------------------------------|
| <a href="#"><u>setres</u></a>    | Create a <a href="#"><u>user reservation</u></a> .                                                 |
| <a href="#"><u>showbf</u></a>    | Shows resource availability for jobs with specific resource requirements.                          |
| <a href="#"><u>showq</u></a>     | Displays detailed prioritized list of active and idle jobs.                                        |
| <a href="#"><u>showstart</u></a> | Shows estimated start time of idle jobs.                                                           |
| <a href="#"><u>showstats</u></a> | Shows detailed usage statistics for users, groups, and accounts, to which the end-user has access. |

#### Related topics

- [Commands Overview](#)

## Commands

### checkjob

#### Synopsis

checkjob [[exact:jobid](#)] [[-l policylevel](#)] [[-n nodeid](#)] [[-q qosid](#)] [[-r reservationid](#)] [[-v](#)] [[--flags=future](#)] [[--blocking](#)] [jobid](#)

#### Overview

checkjob displays detailed job [state](#) information and diagnostic output for a specified job. Detailed information is available for queued, blocked, active, and recently completed jobs. The checkjob command shows the master job of an array as well as a summary of array sub-jobs, but does not display all sub-jobs. Use [checkjob -v](#) to display all job-array sub-jobs.

#### Access

This command can be run by level 1-3 Moab administrators for any job. Also, end users can use checkjob to view the status of their own jobs.

## Arguments

| --blocking         |                                                                                                                   |
|--------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | --blocking                                                                                                        |
| <b>Description</b> | Do not use cache information in the output. The --blocking flag retrieves results exclusively from the scheduler. |
| <b>Example</b>     | <pre>&gt; checkjob -v --blocking 1234</pre> <p><i>Display real time data about job 1234.</i></p>                  |

| --flags            |                                                                                                                                                                     |
|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | --flags=future                                                                                                                                                      |
| <b>Description</b> | Evaluates future eligibility of job (ignore current resource state and usage limitations).                                                                          |
| <b>Example</b>     | <pre>&gt; checkjob -v --flags=future 6235</pre> <p><i>Display reasons why idle job is blocked ignoring node state and current node utilization constraints.</i></p> |

| exact              |                                                   |
|--------------------|---------------------------------------------------|
| <b>Format</b>      | exact:<JOBID>                                     |
| <b>Description</b> | Searches for and returns the exact job ID         |
| <b>Example</b>     | <pre>&gt; checkjob exact:1.job_ dependency1</pre> |

| -l (Policy level) |                                            |
|-------------------|--------------------------------------------|
| <b>Format</b>     | <POLICYLEVEL><br><i>HARD, SOFT, or OFF</i> |

| -l (Policy level)  |                                                                             |
|--------------------|-----------------------------------------------------------------------------|
| <b>Description</b> | Reports job start eligibility subject to specified throttling policy level. |
| <b>Example</b>     | <pre>&gt; checkjob -l SOFT 6235 &gt; checkjob -l HARD 6235</pre>            |

| -n (NodeID)        |                                                                                                                      |
|--------------------|----------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <NODEID>                                                                                                             |
| <b>Description</b> | Checks job access to specified node and <a href="#">preemption</a> status with regards to jobs located on that node. |
| <b>Example</b>     | <pre>&gt; checkjob -n node113 6235</pre>                                                                             |

| -q (QoS)           |                                             |
|--------------------|---------------------------------------------|
| <b>Format</b>      | <QOSID>                                     |
| <b>Description</b> | Checks job access to specified QoS <QOSID>. |
| <b>Example</b>     | <pre>&gt; checkjob -q special 6235</pre>    |

| -r (Reservation)   |                                                     |
|--------------------|-----------------------------------------------------|
| <b>Format</b>      | <RSVID>                                             |
| <b>Description</b> | Checks job access to specified reservation <RSVID>. |
| <b>Example:</b>    | <pre>&gt; checkjob -r orion.1 6235</pre>            |

**-v (Verbose)**

|                    |                                                                                                                                                                                                                                                                                                                                   |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Description</b> | Sets verbose mode. If the job is part of an array, the <code>-v</code> option shows pertinent array information before the job-specific information (see <a href="#">Example 2</a> and <a href="#">Example 3</a> for differences between standard output and <code>-v</code> output).                                             |
|                    | <div style="background-color: #e6f2ff; border-radius: 5px; padding: 5px;"> <span style="font-size: 1.2em; font-weight: bold; color: #4a7c9c;">i</span> Specifying the double verbose (<code>-v -v</code>) displays additional information about the job. See <a href="#">the Output table</a> for details.                 </div> |
| <b>Example</b>     | <pre>&gt; checkjob -v 6235</pre>                                                                                                                                                                                                                                                                                                  |

## Details

This command allows any Moab administrator to check the detailed status and resource requirements of an active, queued, or recently [completed](#) job. Additionally, this command performs numerous diagnostic checks and determines if and where the job could potentially run. Diagnostic checks include [policy](#) violations, reservation constraints, preemption status, and job to resource mapping. If a job cannot run, a text reason is provided along with a summary of how many nodes are and are not available. If the `-v` flag is specified, a node by node summary of resource availability will be displayed for idle jobs.

### Job Eligibility

If a job cannot run, a text reason is provided along with a summary of how many nodes are and are not available. If the `-v` flag is specified, a node by node summary of resource availability will be displayed for idle jobs. For job level eligibility issues, one of the following reasons will be given:

| Reason                                     | Description                                                                     |
|--------------------------------------------|---------------------------------------------------------------------------------|
| <b>job has hold in place</b>               | one or more job holds are currently in place                                    |
| <b>insufficient idle procs</b>             | there are currently not adequate processor resources available to start the job |
| <b>idle procs do not meet requirements</b> | adequate idle processors are available but these do not meet job requirements   |
| <b>start date not reached</b>              | job has specified a minimum <i>start date</i> which is still in the future      |
| <b>expected state is not idle</b>          | job is in an unexpected state                                                   |
| <b>state is not idle</b>                   | job is not in the idle state                                                    |

| Reason                       | Description                                         |
|------------------------------|-----------------------------------------------------|
| <b>dependency is not met</b> | job depends on another job reaching a certain state |
| <b>rejected by policy</b>    | job start is prevented by a throttling policy       |

If a job cannot run on a particular node, one of the following 'per node' reasons will be given:

| Reason          | Description                                      |
|-----------------|--------------------------------------------------|
| <b>Class</b>    | Node does not allow required job class/queue     |
| <b>CPU</b>      | Node does not possess required processors        |
| <b>Disk</b>     | Node does not possess required local disk        |
| <b>Features</b> | Node does not possess required node features     |
| <b>Memory</b>   | Node does not possess required real memory       |
| <b>Network</b>  | Node does not possess required network interface |
| <b>State</b>    | Node is not Idle or Running                      |

## Reservation Access

The `-r` flag can be used to provide detailed information about job access to a specific reservation

## Preemption Status

If a job is marked as a [preemptor](#) and the `-v` and `-n` flags are specified, `checkjob` will perform a job by job analysis for all jobs on the specified node to determine if they can be preempted.

## Output

The `checkjob` command displays the following job attributes:

| Attribute      | Value    | Description                         |
|----------------|----------|-------------------------------------|
| <b>Account</b> | <STRING> | Name of account associated with job |

| Attribute                           | Value                                                   | Description                                                                                                                                                                             |
|-------------------------------------|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Actual Run Time</b>              | [[[DD:]HH:]MM:]SS                                       | Length of time job actually ran.<br> This info is only displayed in simulation mode.                   |
| <b>Allocated Nodes</b>              | Square bracket delimited list of node and processor ids | List of nodes and processors allocated to job                                                                                                                                           |
| <b>Applied Nodeset**</b>            | <STRING>                                                | Nodeset used for job's node allocation                                                                                                                                                  |
| <b>Arch</b>                         | <STRING>                                                | Node architecture required by job                                                                                                                                                       |
| <b>Attr</b>                         | square bracket delimited list of job attributes         | Job Attributes (i.e. [BACKFILL] [PREEMPT])                                                                                                                                              |
| <b>Available Memory**</b>           | <INTEGER>                                               | The available memory requested by job. Moab displays the relative or exact value by returning a comparison symbol (>, <, >=, <=, or ==) with the value (i.e. Available Memory <= 2048). |
| <b>Available Swap**</b>             | <INTEGER>                                               | The available swap requested by job. Moab displays the relative or exact value by returning a comparison symbol (>, <, >=, <=, or ==) with the value (i.e. Available Swap >= 1024).     |
| <b>Average Utilized Procs*</b>      | <FLOAT>                                                 | Average load balance for a job                                                                                                                                                          |
| <b>Avg Util Resources Per Task*</b> | <FLOAT>                                                 |                                                                                                                                                                                         |
| <b>BecameEligible</b>               | <TIMESTAMP>                                             | The date and time when the job moved from Blocked to Eligible.                                                                                                                          |
| <b>Bypass</b>                       | <INTEGER>                                               | Number of times a lower priority job with a later submit time ran before the job                                                                                                        |
| <b>CheckpointStartTime**</b>        | [ [ DD: ] HH: ] MM: ] SS                                | The time the job was first checkpointed                                                                                                                                                 |

| Attribute                              | Value                                             | Description                                                                                                                                                             |
|----------------------------------------|---------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Class</b>                           | [<CLASS NAME><br><CLASS COUNT>]                   | Name of class/queue required by job and number of class initiators required per task.                                                                                   |
| <b>Dedicated Resources Per Task*</b>   | Space-delimited list of<br><STRING><br>:<INTEGER> | Resources dedicated to a job on a per-task basis                                                                                                                        |
| <b>Disk</b>                            | <INTEGER>                                         | Amount of local disk required by job (in MB)                                                                                                                            |
| <b>Estimated Walltime</b>              | [[[DD:]HH:]MM:]SS                                 | The scheduler's estimated walltime.<br> In simulation mode, it is the actual walltime. |
| <b>EnvVariables**</b>                  | Comma-delimited list of <STRING>                  | List of environment variables assigned to job                                                                                                                           |
| <b>Exec Size*</b>                      | <INTEGER>                                         | Size of job executable (in MB)                                                                                                                                          |
| <b>Executable</b>                      | <STRING>                                          | Name of command to run                                                                                                                                                  |
| <b>Features</b>                        | Square bracket delimited list of <STRING>s        | Node features required by job                                                                                                                                           |
| <b>Flags</b>                           |                                                   |                                                                                                                                                                         |
| <b>Group</b>                           | <STRING>                                          | Name of UNIX group associated with job                                                                                                                                  |
| <b>Holds</b>                           | Zero or more of User, System, and Batch           | Types of job holds currently applied to job                                                                                                                             |
| <b>Image Size</b>                      | <INTEGER>                                         | Size of job data (in MB)                                                                                                                                                |
| <b>IWD (Initial Working Directory)</b> | <DIR>                                             | Directory to run the executable in                                                                                                                                      |
| <b>Job Messages**</b>                  | <STRING>                                          | Messages attached to a job                                                                                                                                              |
| <b>Job Submission**</b>                | <STRING>                                          | Job script submitted to RM                                                                                                                                              |

| Attribute                           | Value                                                   | Description                                                                                                                                                                                                                                                                                                                   |
|-------------------------------------|---------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Memory</b>                       | <INTEGER>                                               | Amount of real memory required per node (in MB)                                                                                                                                                                                                                                                                               |
| <b>Max Util Resources Per Task*</b> | <FLOAT>                                                 |                                                                                                                                                                                                                                                                                                                               |
| <b>NodeAccess*</b>                  |                                                         |                                                                                                                                                                                                                                                                                                                               |
| <b>Nodecount</b>                    | <INTEGER>                                               | Number of nodes required by job                                                                                                                                                                                                                                                                                               |
| <b>Opsys</b>                        | <STRING>                                                | Node operating system required by job                                                                                                                                                                                                                                                                                         |
| <b>Partition Mask</b>               | ALL or colon delimited list of partitions               | List of <a href="#">partitions</a> the job has access to                                                                                                                                                                                                                                                                      |
| <b>PE</b>                           | <FLOAT>                                                 | Number of processor-equivalents requested by job                                                                                                                                                                                                                                                                              |
| <b>Per Partition Priority**</b>     | Tabular                                                 | Table showing job template priority for each partition                                                                                                                                                                                                                                                                        |
| <b>Priority Analysis**</b>          | Tabular                                                 | Table showing how job's priority was calculated:<br>Job PRIORITY* Cred( User:Group:Class) Serv (QTime)                                                                                                                                                                                                                        |
| <b>QOS</b>                          | <STRING>                                                | Quality of Service associated with job                                                                                                                                                                                                                                                                                        |
| <b>Reservation</b>                  | <RSVID> ( <TIME1> - <TIME2> Duration: <TIME3>)          | RESID specifies the reservation id, TIME1 is the relative start time, TIME2 the relative end time, TIME3 the duration of the reservation                                                                                                                                                                                      |
| <b>Req</b>                          | [<INTEGER>] TaskCount: <INTEGER> Partition: <partition> | A <a href="#">job requirement</a> for a single type of resource followed by the number of tasks instances required and the appropriate <a href="#">partition</a>                                                                                                                                                              |
| <b>StageIn</b>                      | <SOURCE> %<DESTINATION>                                 | The <SOURCE> is the username, hostname, directory and file name of origin for the file(s) that Moab will stage in for this job. The <DESTINATION> is the username, hostname, directory and file name where Moab will place the file during this job. See <a href="#">About data staging on page 901</a> for more information. |

| Attribute                      | Value                                                                                                   | Description                                                                                                                                                                                                                                                                                                                    |
|--------------------------------|---------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>StageInSize</b>             | <INTEGER><UNIT>                                                                                         | The size of the file Moab will stage in for this job. <UNIT> can be KB, MB, GB, or TB. See <a href="#">About data staging on page 901</a> for more information.                                                                                                                                                                |
| <b>StageOut</b>                | <SOURCE><br>%<DESTINATION>                                                                              | The <SOURCE> is the username, hostname, directory and file name of origin for the file(s) that Moab will stage out for this job. The <DESTINATION> is the username, hostname, directory and file name where Moab will place the file during this job. See <a href="#">About data staging on page 901</a> for more information. |
| <b>StageOutSize</b>            | <INTEGER><UNIT>                                                                                         | The size of the file Moab will stage out for this job. <UNIT> can be KB, MB, GB, or TB. See <a href="#">About data staging on page 901</a> for more information.                                                                                                                                                               |
| <b>StartCount</b>              | <INTEGER>                                                                                               | Number of times job has been started by Moab                                                                                                                                                                                                                                                                                   |
| <b>StartPriority</b>           | <INTEGER>                                                                                               | Start priority of job                                                                                                                                                                                                                                                                                                          |
| <b>StartTime</b>               | <TIME>                                                                                                  | Time job was started by the resource management system                                                                                                                                                                                                                                                                         |
| <b>State</b>                   | One of Idle, Starting, Running, etc. See <a href="#">Job States on page 99</a> for all possible values. | Current Job State                                                                                                                                                                                                                                                                                                              |
| <b>SubmitTime</b>              | <TIME>                                                                                                  | Time job was submitted to resource management system                                                                                                                                                                                                                                                                           |
| <b>Swap</b>                    | <INTEGER>                                                                                               | Amount of swap disk required by job (in MB)                                                                                                                                                                                                                                                                                    |
| <b>Task Distribution*</b>      | Square bracket delimited list of nodes                                                                  |                                                                                                                                                                                                                                                                                                                                |
| <b>Time Queued</b>             |                                                                                                         |                                                                                                                                                                                                                                                                                                                                |
| <b>Total Requested Nodes**</b> | <INTEGER>                                                                                               | Number of nodes the job requested                                                                                                                                                                                                                                                                                              |
| <b>Total Requested Tasks</b>   | <INTEGER>                                                                                               | Number of tasks requested by job                                                                                                                                                                                                                                                                                               |

| Attribute                           | Value                                  | Description                                                    |
|-------------------------------------|----------------------------------------|----------------------------------------------------------------|
| <b>User</b>                         | <STRING>                               | Name of user submitting job                                    |
| <b>Utilized Resources Per Task*</b> | <FLOAT>                                |                                                                |
| <b>WallTime</b>                     | [[[DD:]HH:]MM:]SS of [[[DD:]HH:]MM:]SS | Length of time job has been running out of the specified limit |

In the above table, fields marked with an asterisk (\*) are only displayed when set or when the `-v` flag is specified. Fields marked with two asterisks (\*\*) are only displayed when set or when the `-v -v` flag is specified.

*Example 3-10: checkjob 717*

```

> checkjob 717
job 717
State: Idle
Creds: user:jacksond group:jacksond class:batch
WallTime: 00:00:00 of 00:01:40
SubmitTime: Mon Aug 15 20:49:41
  (Time Queued Total: 3:12:23:13 Eligible: 3:12:23:11)
TerminationDate: INFINITY Sat Oct 24 06:26:40
Total Tasks: 1
Req[0] TaskCount: 1 Partition: ALL
Network: --- Memory >= 0 Disk >= 0 Swap >= 0
Opsys: --- Arch: --- Features: ---

IWD: /home/jacksond/moab/moab-4.2.3
Executable: STDIN
Flags: RESTARTABLE,NORMSTART
StartPriority: 5063
Reservation '717' ( INFINITY -> INFINITY Duration: 00:01:40)
Note: job cannot run in partition base (idle procs do not meet requirements : 0 of 1
procs found)
idle procs: 4 feasible procs: 0
Rejection Reasons: [State : 3][ReserveTime : 1]
cannot select job 717 for partition GM (partition GM does not support requested class
batch)

```

*The example job cannot be started for two different reasons.*

- It is temporarily blocked from partition *base* because of node state and node reservation conflicts.
- It is permanently blocked from partition *GM* because the requested class *batch* is not supported in that partition.

**Example 3-11: Using `checkjob` (no `-v`) on a job array master job:**

```

checkjob array.1
job array.1

AName: array
Job Array Info:
  Name: array.1

Sub-jobs:          10
Active:            6 ( 60.0%)
Eligible:          2 ( 20.0%)
Blocked:           2 ( 20.0%)
Complete:          0 (  0.0%)

```

**Example 3-12: Using `checkjob -v` on a job array master job:**

```

$ checkjob -v array.1
job array.1

AName: array
Job Array Info:
  Name: array.1
  1 : array.1.1 : Running
  2 : array.1.2 : Running
  3 : array.1.3 : Running
  4 : array.1.4 : Running
  5 : array.1.5 : Running
  6 : array.1.6 : Running
  7 : array.1.7 : Idle
  8 : array.1.8 : Idle
  9 : array.1.9 : Blocked
 10 : array.1.10 : Blocked

Sub-jobs:          10
Active:            6 ( 60.0%)
Eligible:          2 ( 20.0%)
Blocked:           2 ( 20.0%)
Complete:          0 (  0.0%)

```

**Example 3-13: Using `checkjob -v` on a data staging job**

```

$ checkjob -v moab.14.dsin
job moab.14.dsin

AName: moab.14.dsin
State: Running
Creds: user:fred group:company
WallTime: 00:00:00 of 00:01:01
SubmitTime: Wed Apr 16 10:07:19
      (Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Wed Apr 16 10:07:19
TemplateSets: dsin
Triggers: 78$start+0@0.000000:exec@/opt/moab/tools/datastaging/ds_move_rsync --
stagein:FALSE
Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: SHARED
Dedicated Resources Per Task: bandwidth: 1
NodeAccess: SHARED

Allocated Nodes:
[GLOBAL:1]

Job Group: moab.14
SystemID: moab
SystemJID: moab.14.dsin
Task Distribution: GLOBAL
IWD: $HOME/test/datastaging
SubmitDir: $HOME/test/datastaging
StartCount: 1
Parent VCs: vc11
User Specified Partition List: local
Partition List: local
SrcRM: internal
Flags: NORMSTART,GRESONLY,TEMPLATESAPPLIED
Attr: dsin
StageInSize: 386MB
StageOutSize: 100MB
StageIn: fred@remotelab:/home/fred/input1/%fred@scratch:/home/fred/input1/
StageIn: fred@remotelab:/home/fred/input2/%fred@scratch:/home/fred/input2/
StageIn: fred@remotelab:/home/fred/input3/%fred@scratch:/home/fred/input3/
StageOut: fred@scratch:/home/fred/output/%fred@remotelab:/home/fred/output/
StartPriority: 1
  SJob Type: datastaging
  Completion Policy: datastaging
PE: 0.00
Reservation 'moab.14.dsin' (-00:00:06 -> 00:00:55 Duration: 00:01:01)

```

**Related topics**

- [showhist.moab.pl](#) - explains how to query for past job information
- Moab Client Installation - explains how to distribute this command to client nodes
- [mdiag -j](#) command - display additional detailed information regarding jobs
- [showq](#) command - showq high-level job summaries
- [JOBPURGETIME](#) parameter - specify how long information regarding completed jobs is maintained
- diagnosing job [preemption](#)

# checknode

## Synopsis

```
checknode options nodeID
          ALL
```

## Overview

This command shows detailed state information and statistics for nodes that run jobs.

The following information is returned by this command:

| Name             | Description                                                                                             |
|------------------|---------------------------------------------------------------------------------------------------------|
| <b>Disk</b>      | Disk space available                                                                                    |
| <b>Memory</b>    | Memory available                                                                                        |
| <b>Swap</b>      | Swap space available                                                                                    |
| <b>State</b>     | Node state                                                                                              |
| <b>Opsys</b>     | Operating system                                                                                        |
| <b>Arch</b>      | Architecture                                                                                            |
| <b>Adapters</b>  | Network adapters available                                                                              |
| <b>Features</b>  | Features available                                                                                      |
| <b>Classes</b>   | Classes available                                                                                       |
| <b>StateTime</b> | Time node has been in current state in HH:MM:SS notation                                                |
| <b>Downtime</b>  | Displayed only if downtime is scheduled                                                                 |
| <b>Load</b>      | CPU Load (Berkley one-minute load average)                                                              |
| <b>TotalTime</b> | Total time node has been detected since statistics initialization expressed in <i>HH:MM:SS</i> notation |

| Name                       | Description                                                                                                                                                               |
|----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UpTime</b>              | Total time node has been in an available (Non-Down) state since statistics initialization expressed in <i>HH:MM:SS</i> notation (percent of time up: UpTime/TotalTime)    |
| <b>ActiveTime</b>          | Total time node has been busy (allocated to active jobs) since statistics initialization expressed in <i>HH:MM:SS</i> notation (percent of time busy: BusyTime/TotalTime) |
| <b>EffNodeAccessPolicy</b> | Configured effective node access policy                                                                                                                                   |

After displaying this information, some analysis is performed and any unusual conditions are reported.

## Access

By default, this command can be run by any Moab Administrator (see [ADMINCFG](#)).

## Parameters

| Name        | Description                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>NODE</b> | Node name you want to check. Moab uses regular expressions to return any node that contains the provided argument. For example, if you ran <code>checknode node1</code> , Moab would return information about <code>node1</code> , <code>node10</code> , <code>node100</code> , etc. If you want to limit the results to <code>node1</code> only, you would run <code>checknode "^node1\$"</code> . |

## Flags

| Name         | Description                                                    |
|--------------|----------------------------------------------------------------|
| <b>ALL</b>   | Returns checknode output on all nodes in the cluster.          |
| <b>-h</b>    | Help for this command.                                         |
| <b>-v</b>    | Returns verbose output.                                        |
| <b>--xml</b> | Output in XML format. Same as <a href="#">mdiag -n --xml</a> . |

*Example 3-14: checknode*

```
> checknode P690-032
node P690-032

State:      Busy (in current state for 11:31:10)
Configured Resources: PROCS: 1 MEM: 16G SWAP: 2000M DISK: 500G
Utilized Resources: PROCS: 1
Dedicated Resources: PROCS: 1
Opsys:     AIX      Arch:      P690
Speed:     1.00     CPUload:  1.000
Network:   InfiniBand,Myrinet
Features:  Myrinet
Attributes: [Batch]
Classes:   [batch]

Total Time: 5:23:28:36 Up: 5:23:28:36 (100.00%) Active: 5:19:44:22 (97.40%)

Reservations:
  Job '13678'(x1) 10:16:12:22 -> 12:16:12:22 (2:00:00:00)
  Job '13186'(x1) -11:31:10 -> 1:12:28:50 (2:00:00:00)
Jobs: 13186
```

*Example 3-15: checknode ALL*

```

> checknode ALL
node ahe

State:      Idle (in current state for 00:00:30)
Configured Resources: PROCS: 12 MEM: 8004M SWAP: 26G DISK: 1M
Utilized Resources: PROCS: 1 SWAP: 4106M
Dedicated Resources: ---
  MTBF(longterm): INFINITY MTBF(24h): INFINITY
Opsys:      linux Arch: ---
Speed:      1.00 CPUload: 1.400
Flags:      rmdetected
Classes:    [batch]
RM[ahe]*:   TYPE=PBS
EffNodeAccessPolicy: SHARED

Total Time: 00:01:44 Up: 00:01:44 (100.00%) Active: 00:00:00 (0.00%)

Reservations: ---
node ahe-ubuntu32

State:      Running (in current state for 00:00:05)
Configured Resources: PROCS: 12 MEM: 2013M SWAP: 3405M DISK: 1M
Utilized Resources: PROCS: 6 SWAP: 55M
Dedicated Resources: PROCS: 6
  MTBF(longterm): INFINITY MTBF(24h): INFINITY
Opsys:      linux Arch: ---
Speed:      1.00 CPUload: 2.000
Flags:      rmdetected
Classes:    [batch]
RM[ahe]*:   TYPE=PBS
EffNodeAccessPolicy: SHARED

Total Time: 00:01:44 Up: 00:01:44 (100.00%) Active: 00:00:02 (1.92%)

Reservations:
  6x2 Job:Running -00:00:07 -> 00:01:53 (00:02:00)
  7x2 Job:Running -00:00:06 -> 00:01:54 (00:02:00)
  8x2 Job:Running -00:00:05 -> 00:01:55 (00:02:00)
Jobs:      6,7,8
node ahe-ubuntu64

State:      Busy (in current state for 00:00:06)
Configured Resources: PROCS: 12 MEM: 2008M SWAP: 3317M DISK: 1M
Utilized Resources: PROCS: 12 SWAP: 359M
Dedicated Resources: PROCS: 12
  MTBF(longterm): INFINITY MTBF(24h): INFINITY
Opsys:      linux Arch: ---
Speed:      1.00 CPUload: 0.000
Flags:      rmdetected
Classes:    [batch]
RM[ahe]*:   TYPE=PBS
EffNodeAccessPolicy: SHARED

Total Time: 00:01:44 Up: 00:01:44 (100.00%) Active: 00:00:55 (52.88%)

Reservations:
  0x2 Job:Running -00:01:10 -> 00:00:50 (00:02:00)
  1x2 Job:Running -00:00:20 -> 00:01:40 (00:02:00)
  2x2 Job:Running -00:00:20 -> 00:01:40 (00:02:00)
  3x2 Job:Running -00:00:17 -> 00:01:43 (00:02:00)
  4x2 Job:Running -00:00:13 -> 00:01:47 (00:02:00)
  5x2 Job:Running -00:00:07 -> 00:01:53 (00:02:00)

```

```
Jobs:          0,1,2,3,4,5
ALERT: node is in state Busy but load is low (0.000)
```

Related topics

- Moab Client Installation - explains how to distribute this command to client nodes
- [mdiag -n](#)
- [showstate](#)

mcredctl

Synopsis

```
mcredctl [-d credtype[:credid]] [-h credtype:credid] [-l credtype] [-q
{role|limit|profile|accessfrom|accesssto|policies} credtype[:credid]] [--format=xml] [-r
{stats|credits|fairshare} credtype[:credid]] [-t <STARTTIME>[,<ENDTIME>]
```

Overview

The mcredctl command controls various aspects about the credential objects within Moab. It can be used to display configuration, limits, roles, and relationships for various Moab credential objects.

Arguments

**i** In all cases <CREDTYPE> is one of acct, group, user, class, or qos.

**i** In most cases it is necessary to use the --format=xml flag in order to print the output (see examples below for specific syntax requirements).

| -d - DESTROY       |                                                                                                                               |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <TYPE>: <VAL>                                                                                                                 |
| <b>Description</b> | Purge a credential from moab.cfg (does not delete credential from memory).                                                    |
| <b>Example</b>     | <pre>&gt; mcredctl -d user:john</pre> <p><i>All references to <b>USERCFG</b>[john] will be commented out of moab.cfg)</i></p> |

| -h - HOLD          |                                                                                      |
|--------------------|--------------------------------------------------------------------------------------|
| <b>Format</b>      | <TYPE>:<VAL>                                                                         |
| <b>Description</b> | Toggles whether a given credentials' jobs should be place on hold or not.            |
| <b>Example</b>     | <pre>&gt; mcredctl -h user:john</pre> <p><i>User [john] will be put on hold.</i></p> |

| -l - LIST          |                                                                                                                                                                                                 |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <TYPE>                                                                                                                                                                                          |
| <b>Description</b> | List the various sub-objects of the specified credential.                                                                                                                                       |
| <b>Example</b>     | <pre>&gt; mcredctl -l user --format=xml</pre> <p><i>List all users within Moab in XML.</i></p> <pre>&gt; mcredctl -l group --format=xml</pre> <p><i>List all groups within Moab in XML.</i></p> |

| -q - QUERY          |                                                                                                                                                                                                          |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>       | {role   accessfrom   accessto   limit   profile   policies}<br>limit <TYPE><br>policies <TYPE><br>role <USER>:<USERID><br>profile <TYPE>[:<VAL>]<br>accessfrom <TYPE>[:<VAL>]<br>accessto <TYPE>[:<VAL>] |
| <b>Description:</b> | Display various aspects of a credential (formatted in XML)                                                                                                                                               |

**-q - QUERY**

|                 |                                                                                                |
|-----------------|------------------------------------------------------------------------------------------------|
| <b>Example:</b> | <pre>&gt; mcredctl -q role user:bob --format=xml</pre>                                         |
|                 | <i>View user bob's administrative role within Moab in XML</i>                                  |
|                 | <pre>&gt; mcredctl -q limit acct --format=xml</pre>                                            |
|                 | <i>Display limits for all accounts in XML</i>                                                  |
|                 | <pre>&gt; mcredctl -q policies user:bob</pre>                                                  |
|                 | <i>View limits organized by credential for user bob on each partition and resource manager</i> |

**-r - RESET**

|                    |                                                                                    |
|--------------------|------------------------------------------------------------------------------------|
| <b>Format</b>      | <TYPE>                                                                             |
| <b>Description</b> | Resets the credential within Moab.                                                 |
| <b>Example</b>     | <pre>&gt; mcredctl -r user:john</pre><br><i>Resets the credential of user john</i> |

**-t - TIMEFRAME**

|                    |                                                                                                                     |
|--------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Format</b>      | <STARTTIME>[,<ENDTIME>]                                                                                             |
| <b>Description</b> | Can be used in conjunction with the -q profile option to display profiling information for the specified timeframe. |
| <b>Example</b>     | <pre>&gt; mcredctl -q profile user -t 14:30_06/20</pre>                                                             |

### Credential Statistics XML Output

Credential statistics can be requested as XML (via the --format=xml argument) and will be written to STDOUT in the following format:

```
> mcredctl -q profile user --format=xml -o time:1182927600,1183013999
<Data>
  <user ...>
    <Profile ...>
    </Profile>
  </user>
</Data>
```

#### Example 3-16: Deleting a group

```
> mcredctl -d group:john
GROUPCFG[john] Successfully purged from config files
```

#### Example 3-17: List users in XML format

```
> mcredctl -l user --format=xml
<Data><user ID="john"></user><user ID="john"></user><user ID="root"></user><user
ID="dev"></user></Data>
```

#### Example 3-18: Display information about a user

```
> mcredctl -q role user:john --format=xml
<Data><user ID="test" role="admin5"></user></Data>
```

### Related topics

- Moab Client Installation - explains how to distribute this command to client nodes

## mdiag

### Synopsis

[mdiag -a](#) [*accountid*]

[mdiag -b](#) [-l *policylevel*] [-t *partition*]

[mdiag -c](#) [*classid*]

[mdiag -C](#) [*configfile*] // diagnose config file syntax

[mdiag -e](#) [-w *starttime*]|*endtime*|*eventtypes*|*oidlist*|*eidlist*|*objectlist*] --xml

[mdiag -f](#) [-o user|group|acct|qos|class] [-v]

[mdiag -g](#) [*groupid*]

[mdiag -G](#) [*Green*]

[mdiag -j](#) [*jobid*] [-t *partition*] [-v] [--blocking]

[mdiag -L](#) [-v] // diagnose usage limits

[mdiag -n](#) [-A *creds*] [-t *partition*] [*nodeid*] [-v]

[mdiag -p](#) [-t *partition*] [-v] // diagnose job priority

[mdiag -q](#) [*qosid*]

`mdiag -r [reservationid] [-v] [-w type=<type>] [--blocking]`

`mdiag -R [resourcemanagename] [-v]`

`mdiag -s [standingreservationid] [--blocking]`

`mdiag -S [-v] // diagnose scheduler`

`mdiag -t [-v] // diagnose partitions`

`mdiag -T [triggerid] [-v] [--blocking]`

`mdiag -u [userid]`

`mdiag [--format=xml]`

## Overview

The `mdiag` command is used to display information about various aspects of the cluster and the results of internal diagnostic tests. In summary, it provides the following:

- current object health and state information
- current object configuration (resources, policies, attributes, etc)
- current and historical performance/utilization information
- reports on recent failure
- object messages

Some `mdiag` options gather information from the Moab cache which prevents them from interrupting the scheduler, but the `--blocking` option can be used to bypass the cache and interrupt the scheduler.

## Arguments

| Argument                    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-a [accountid]</code> | Display <code>account</code> information                                                                                                                                                                                                                                                                                                                                                                                                      |
| <code>-b</code>             | Display information on jobs blocked by policies, holds, or other factors.<br><div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> If blocked job diagnostics are specified, the <code>-t</code> option is also available to constrain the report to analysis of particular partition. Also, with blocked job diagnosis, the <code>-l</code> option can be used to specify the analysis policy level.</p> </div> |
| <code>-c [classid]</code>   | Display <code>class</code> information                                                                                                                                                                                                                                                                                                                                                                                                        |

| Argument            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-C [file]</b>    | With the vast array of options in the configuration file, the <code>-C</code> option does not validate function, but it does analyze the configuration file for syntax errors including use of invalid parameters, deprecated parameters, and some illegal values. If you start Moab with the <code>-e</code> flag, Moab evaluates the configuration file at startup and quits if an error exists.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>-e</b>           | Moab will do a query for all events whose <i>eventtime</i> starts at <i>&lt;starttime&gt;</i> and matches the search criteria. This works only when Moab is configured with ODBC MySQL. The syntax is:<br><code>mdiag -e[-w &lt;starttime&gt; &lt;eventtypes&gt; &lt;oidlist&gt; &lt;eidlist&gt; &lt;objectlist&gt;] --xml</code> <ul style="list-style-type: none"> <li>• <i>starttime</i> default is -</li> <li>•</li> <li>• <i>eventtypes</i> default is command delimited, the default is all event types (possible values can be found in the EventType table in the Moab database)</li> <li>• <i>oidlist</i> is a comma-delimited list of object ids, the default is all objects ids</li> <li>• <i>eidlist</i> is a comma-delimited list of specific event ids, the default is all event ids</li> <li>• <i>objectlist</i> is a comma-delimited list of object types, the default is all object types (possible values can be found in the ObjectType table in the Moab database)</li> </ul> |
| <b>-f</b>           | Display <a href="#">fairshare</a> information                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>-g [groupid]</b> | display <a href="#">group</a> information                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>-G [Green]</b>   | display power management information                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b>-j [jobid]</b>   | display job information                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>-L</b>           | display limits                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>-n [nodeid]</b>  | display nodes<br><div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  If node diagnostics are specified, the <code>-t</code> option is also available to constrain the report to a particular partition. </div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b>-p</b>           | display <a href="#">job priority</a> .<br><div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  If priority diagnostics are specified, the <code>-t</code> option is also available to constrain the report to a particular partition. </div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>-q [qosid]</b>   | display <a href="#">qos</a> information                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

| Argument                                    | Description                                                                                          |
|---------------------------------------------|------------------------------------------------------------------------------------------------------|
| <b>-r</b> [ <a href="#">reservationid</a> ] | display reservation information                                                                      |
| <b>-R</b> [ <a href="#">rmid</a> ]          | display resource manager information                                                                 |
| <b>-s</b> [ <a href="#">srsv</a> ]          | display <a href="#">standing reservation</a> information                                             |
| <b>-S</b>                                   | display general scheduler information                                                                |
| <b>-t</b>                                   | display configuration, usage, health, and diagnostic information about partitions maintained by Moab |
| <b>-T</b> [ <a href="#">triggerid</a> ]     | display trigger information                                                                          |
| <b>-u</b> [ <a href="#">userid</a> ]        | display <a href="#">user</a> information                                                             |
| <b>--format=xml</b>                         | display output in XML format                                                                         |

## XML Output

Information for most of the options can be reported as XML as well. This is done with the command `mdiag -<option> <CLASS_ID> --format=xml`. For example, XML-based class information will be written to STDOUT in the following format:

```
<Data>
  <class <ATTR>="<VAL>" ... >
    <stats <ATTR>="<VAL>" ... >
      <Profile <ATTR>="<VAL>" ... >
        </Profile>
      </stats>
    </class>
  <Data>
  ...
</Data>
```

Of the `mdiag` options, only `-G` and `-L` cannot be reported as XML.

## Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [checkjob](#)
- [checknode](#)

## mdiag -a

## Synopsis

```
mdiag -a [accountid]
```

## Overview

The `mdiag -a` command provides detailed information about the [accounts](#) (aka projects) Moab is currently tracking. This command also allows an administrator to verify correct throttling policies and access provided to and from other credentials.

*Example 3-19: Generating information about accounts*

```
> mdiag -a
evaluating acct information
Name      Priority  Flags      QDef      QOSList*
PartitionList Target Limits
engineering 100      -          high     high,urgent,low [A]
[B]      30.00  MAXJOB=50,75  MAXPROC=400,500
marketing  1        -          low      low [A]
          5.00  MAXJOB=100,110  MAXPS=54000,54500
it        10       -          DEFAULT  DEFAULT,high,urgent,low [A]
          100.00  MAXPROC=100,1250  MAXPS=12000,12500
          FSWEIGHT=1000
development 100      -          high     high,urgent,low [A]
[B]      30.00  MAXJOB=50,75  MAXNODE=100,120
research  100     -          high     DEFAULT,high,low [A]
[B]      30.00  MAXNODE=400,500  MAXPS=900000,1000000
DEFAULT   0        -          -        -
          0.00  -
```

### Related topics

- [Account](#) credential

### **mdiag -b**

## Synopsis

```
mdiag -b [-l policylevel] [-t partition]
```

## Overview

The `mdiag -b` command returns information about blocked jobs.

### **mdiag -c**

## Synopsis

```
mdiag -c [-v] [classid]
```

## Overview

The `mdiag -c` command provides detailed information about the classes Moab is currently tracking. This command also allows an administrator to verify correct throttling policies and access provided to and from other credentials.

 The term class is used interchangeably with the term queue and generally refers to resource manager queue.

## XML Attributes

Name	Description
<b>ADEF</b>	Accounts a class has access to.
<b>CAPACITY</b>	Number of procs available to the class.
<b>DEFAULT.ATTR</b>	Default attributes attached to a job.
<b>DEFAULT.DISK</b>	Default required disk attached to a job.
<b>DEFAULT.FEATURES</b>	Default required node features attached to a job.
<b>DEFAULT.GRES</b>	Default generic resources attached to a job.
<b>DEFAULT.MEM</b>	Default required memory attached to a job.
<b>DEFAULT.NODESET</b>	Default specified nodeset attached to a job.
<b>DEFAULT.WCLIMIT</b>	Default wallclock limit attached to a job.
<b>EXCL.FEATURES</b>	List of excluded (disallowed) node features.
<b>EXCL.FLAGS</b>	List of excluded (disallowed) job flags.
<b>FSTARGET</b>	The class' fairshare target.
<b>HOLD</b>	If TRUE this credential has a hold on it, FALSE otherwise.
<b>HOSTLIST</b>	The list of hosts in this class.

Name	Description
<b>JOBEPILOG</b>	Scheduler level job epilog to be run after job is completed by resource manager (script path).
<b>JOBFLAGS</b>	Default flags attached to jobs in the class.
<b>JOBPROLOG</b>	Scheduler level job prolog to be run before job is started by resource manager (script path).
<b>ID</b>	The unique ID of this class.
<b>LOGLEVEL</b>	The log level attached to jobs in the class.
<b>MAX.PROC</b>	The max processors per job in the class.
<b>MAX.PS</b>	The max processor-seconds per job in the class.
<b>MAX.WCLIMIT</b>	The max wallclock limit per job in the class.
<b>MAXIJOB</b>	The max idle jobs in the class.
<b>MAXIPROC</b>	The max idle processors in the class.
<b>MAXJOBPERUSER</b>	The max jobs per user.
<b>MAXNODEPERJOB</b>	The max nodes per job.
<b>MAXNODEPERUSER</b>	The max nodes per user.
<b>MAXPROCPERJOB</b>	The max processors per job.
<b>MAXPROCPERNODE</b>	The max processors per node.
<b>MAXPROCPERUSER</b>	The max processors per user.
<b>MIN.NODE</b>	The minimum nodes per job in the class.
<b>MIN.PROC</b>	The minimum processors per job in the class.

Name	Description
<b>MIN.WCLIMIT</b>	The minimum wallclock limit per job in the class.
<b>NODEACCESSPOLICY</b>	The node access policy associated with jobs in the class.
<b>OCDPROCFACTOR</b>	Dedicated processor factor.
<b>OCNODE</b>	Overcommit node.
<b>PRIORITY</b>	The class' associated priority.
<b>PRIORITYF</b>	Priority calculation function.
<b>REQ.FEATURES</b>	Required features for a job to be considered in the class.
<b>REQ.FLAGS</b>	Required flags for a job to be considered in the class.
<b>REQ.IMAGE</b>	Required image for a job to be considered in the class.
<b>REQUIREDUSERLIST</b>	The list of users who have access to the class.
<b>RM</b>	The resource manager reporting the class.
<b>STATE</b>	The class' state.
<b>WCOVERRUN</b>	Tolerated amount of time beyond the specified wallclock limit.

## Example 3-20: Generating information about classes

```

> mdiag -c
Class/Queue Status
ClassID      Priority  Flags      QDef          QOSList*  PartitionList
Target Limits
DEFAULT      0      ---      ---          ---      ---
0.00      ---
batch        1      ---      ---          ---      [A] [B]
70.00      MAXJOB=33:200,250
          MAX.WCLIMIT=10:00:00  MAXPROCPERJOB=128
long         1      ---      low          low      [A]
10.00      MAXJOB=3:100,200
          MAX.WCLIMIT=1:00:00:00  MAXPROCPERJOB=128
fast         100    ---      high         high     [B]
10.00      MAXJOB=8:100,150
          MAX.WCLIMIT=00:30:00  MAXPROCPERJOB=128
bigmem       1      ---      low,high     low      ---
10.00      MAXJOB=1:100,200
          MAXPROCPERJOB=128

```

In the example above, class **fast** has **MAXJOB** soft and hard limits of **100** and **150** respectively and is currently running **8** jobs.

 The Limits column will display limits in the following format:  
<USAGE>:<HARDLIMIT>[,<SOFTLIMIT>]

## Related topics

- [showstats](#) command - display general statistics

**mdiag -f****Synopsis**

```
mdiag -f [-o user|group|acct|qos|class] [--flags=relative] [-w par=<PARTITIONID>]
```

**Overview**

The `mdiag -f` command is used to display at a glance information about the fairshare configuration and historic resource utilization. The fairshare usage may impact job prioritization, job eligibility, or both based on the credential **FSTARGET** and **FSCAP** attributes and by the fairshare priority weights as described in the [Job Prioritization Overview](#). The information presented by this command includes fairshare configuration and credential fairshare usage over time.

The command hides information about credentials which have no fairshare target and no fairshare cap.

If an object type (<OTYPE>) is specified, then only information for that credential type (user, group, acct, class, or qos) will be displayed. If the `relative` flag is set, then per user fairshare usage will be displayed relative to each non-user credential (see the second example below).

 Relative output is only displayed for credentials which have user mappings. For example, if there is no association between classes and users, no relative per user fairshare usage class breakdown will be provided.

Example 3-21: Standard Fairshare Output

```

> mdiag -f
FairShare Information
Depth: 6 intervals   Interval Length: 00:20:00   Decay Rate: 0.50
FS Policy: DEDICATEDPES
System FS Settings: Target Usage: 0.00
FSInterval      %      Target      0          1          2          3          4          5
FSWeight        -----
TotalUsage      100.00 -----
USER
-----
mattp           2.51 -----   2.20   2.69   2.21   2.65   2.65   3.01
jsmith         12.82 -----   12.66  15.36  10.96  8.74   8.15  13.85
kyliem         3.44 -----   3.93   2.78   4.36   3.11   3.94   4.25
tgh            4.94 -----   4.44   5.12   5.52   3.95   4.66   4.76
walex          1.51 -----   3.14   1.15   1.05   1.61   1.22   1.60
jimf           4.73 -----   4.67   4.31   5.67   4.49   4.93   4.92
poy            4.64 -----   4.43   4.61   4.58   4.76   5.36   4.90
mjackson       0.66 -----   0.35   0.78   0.67   0.77   0.55   0.43
tfw            17.44 -----  16.45  15.59  19.93  19.72  21.38  15.68
gjohn          2.81 -----   1.66   3.00   3.16   3.06   2.41   3.33
ljill          10.85 -----  18.09   7.23  13.28   9.24  14.76   6.67
kbill          11.10 -----   7.31  14.94   4.70  15.49   5.42  16.61
stevei         1.58 -----   1.41   1.34   2.09   0.75   3.30   2.15
gms            1.54 -----   1.15   1.74   1.63   1.40   1.38   0.90
patw           5.11 -----   5.22   5.11   4.85   5.20   5.28   5.78
wer            6.65 -----   5.04   7.03   7.52   6.80   6.43   2.83
anna           1.97 -----   2.29   1.68   2.27   1.80   2.37   2.17
susieb         5.69 -----   5.58   5.55   5.57   6.48   5.83   6.16
GROUP
-----
dallas         13.25  15.00   14.61  12.41  13.19  13.29  15.37  15.09
sanjose*       8.86  15.00   6.54   9.55   9.81   8.97   8.35   4.16
seattle        10.05  15.00   9.66  10.23  10.37   9.15   9.94  10.54
austin*        30.26  15.00  29.10  30.95  30.89  28.45  29.53  29.54
boston*        3.44  15.00   3.93   2.78   4.36   3.11   3.94   4.25
orlando*       26.59  15.00  29.83  26.77  22.56  29.49  25.53  28.18
newyork*       7.54  15.00   6.33   7.31   8.83   7.54   7.34   8.24
ACCT
-----
engineering    31.76  30.00   32.25  32.10  31.94  30.07  30.74  31.14
marketing       8.86   5.00   6.54   9.55   9.81   8.97   8.35   4.16
it              9.12   5.00   7.74   8.65  10.92   8.29  10.64  10.40
development*   24.86  30.00  24.15  24.76  25.00  24.84  26.15  26.78
research        25.40  30.00  29.32  24.94  22.33  27.84  24.11  27.53
QOS
-----
DEFAULT*       0.00  50.00 -----
high*          83.69  90.00 -----  86.76  83.20  81.71  84.35  83.19  88.02
urgent         0.00   5.00 -----
low*          12.00   5.00 -----  7.34  12.70  14.02  12.51  12.86   7.48
CLASS
-----
batch*         51.69  70.00   53.87  52.01  50.80  50.38  48.67  52.65
long*          18.75  10.00   16.54  18.36  20.89  18.36  21.53  16.28
fast*          15.29  10.00   18.41  14.98  12.58  16.80  15.15  18.21
bigmem         14.27  10.00   11.17  14.65  15.73  14.46  14.65  12.87

```

**i** An asterisk (\*) next to a credential name indicates that that credential has exceeded its fairshare target.

**Example 3-22: Grouping User Output by Account**

```

> mdiag -f -o acct --flags=relative
FairShare Information
Depth: 6 intervals   Interval Length: 00:20:00   Decay Rate: 0.50
FS Policy: DEDICATEDPES
System FS Settings:  Target Usage: 0.00
FSInterval          %      Target          0          1          2          3          4          5
FSWeight            -----
TotalUsage          100.00 -----
ACCOUNT
-----
dallas              13.12  15.00    15.42  12.41  13.19  13.29  15.37  15.09
  mattp             19.47  -----    15.00  21.66  16.75  19.93  17.26  19.95
  walex             9.93   -----    20.91  9.28   7.97  12.14  7.91  10.59
  stevei            12.19  -----    9.09  10.78  15.85  5.64  21.46  14.28
  anna              14.77  -----    16.36  13.54  17.18  13.55  15.44  14.37
  susieb           43.64  -----    38.64  44.74  42.25  48.74  37.92  40.81
sanjose*            9.26  15.00    8.69  9.55  9.81  8.97  8.35  4.16
  mjackson          7.71  -----    6.45  8.14  6.81  8.62  6.54  10.29
  gms               17.61  -----    21.77  18.25  16.57  15.58  16.51  21.74
  wer               74.68  -----    71.77  73.61  76.62  75.80  76.95  67.97
seattle            10.12  15.00    10.16  10.23  10.37  9.15  9.94  10.54
  tgh              49.56  -----    46.21  50.05  53.26  43.14  46.91  45.13
  patw             50.44  -----    53.79  49.95  46.74  56.86  53.09  54.87
austin*            30.23  15.00    25.58  30.95  30.89  28.45  29.53  29.54
  jsmith           42.44  -----    48.77  49.62  35.47  30.70  27.59  46.90
  tfw              57.56  -----    51.23  50.38  64.53  69.30  72.41  53.10
boston*            3.38  15.00    3.78  2.78  4.36  3.11  3.94  4.25
  kyliem           100.00 -----    100.00  100.00  100.00  100.00  100.00  100.00
orlando*           26.20  15.00    30.13  26.77  22.56  29.49  25.53  28.18
  poy              17.90  -----    16.28  17.22  20.30  16.15  20.98  17.39
  ljill            37.85  -----    58.60  26.99  58.87  31.33  57.79  23.67
  kbill           44.25  -----    25.12  55.79  20.83  52.52  21.23  58.94
newyork*           7.69  15.00    6.24  7.31  8.83  7.54  7.34  8.24
  jimf             61.42  -----    69.66  58.94  64.20  59.46  67.21  59.64
  gjohn            38.58  -----    30.34  41.06  35.80  40.54  32.79  40.36

```

**Related topics**

- [Fairshare Overview](#)

**mdiag -g****Synopsis**

```
mdiag -g [groupid]
```

**Overview**

The `mdiag -g` command is used to present information about groups.

**mdiag -j****Synopsis**

```
mdiag -j [jobid] [-t <partition>] [-v] [-w] [--flags=policy] [--xml] [--blocking]
```

## Overview

The `mdiag -j` command provides detailed information about the state of jobs Moab is currently tracking. This command also performs a large number of sanity and state checks. The job configuration and status information, as well as the results of the various checks, are presented by this command. The command gathers information from the Moab cache which prevents it from interrupting the scheduler, but the `--blocking` option can be used to bypass the cache and interrupt the scheduler. If the `-v` (verbose) flag is specified, additional information about less common job attributes is displayed. If `--flags=policy` is specified, information about job templates is displayed.

If used with the `-t <partition>` option on a running job, the only thing `mdiag -j` shows is if the job is running on the specified partition. If used on job that is not running, it shows if the job is able to run on the specified partition.

The `-w` flag enables you to specify specific job states (Such as Running, Completed, Idle, or ALL. See [Job States on page 99](#) for all valid options.) or jobs associated with a given credential (user, acct, class, group, qos). For example:

```
mdiag -j -w user=david           # Displays only David's jobs
mdiag -j -w state=Idle,Running  # Displays only idle or running jobs
```

**i** The `mdiag -j` command does not show all subjobs of an array unless you use `mdiag -j --xml`. In the XML, the master job element contains a child element called `ArraySubJobs` that contains the subjobs in the array. Using `mdiag -j -v --xml` shows the completed sub-jobs as well.

## XML Output

If XML output is requested (via the `--format=xml` argument), XML based node information will be written to STDOUT in the following format:

```
<Data>
  <job ATTR="VALUE" ... > </job>
  ...
</Data>
```

For information about legal attributes, refer to the [XML Attributes](#) table.

**i** To show jobs in XML, use `mdiag -j --xml -w [completed=true|system=true|ALL=true]` to limit or filter jobs. This is for XML use only.

### Related topics

- [checkjob](#)
- [mdiag](#)

### mdiag -n

## Synopsis

```
mdiag -n [-t partitionid] [-A creds] [-w <CONSTRAINT>] [-v] [--format=xml] [nodeid]
```

## Overview

The `mdiag -n` command provides detailed information about the state of nodes Moab is currently tracking. This command also performs a large number of sanity and state checks. The node configuration and status information as well as the results of the various checks are presented by this command.

## Arguments

Flag	Argument	Description
<b>[-A]</b>	{user group account qos class job}: <OBJECTID>	report if each node is accessible by requested job or credential
<b>[-t]</b>	<partitionid>	report only nodes from specified partition
<b>[-v]</b>	---	show verbose output (do not truncate columns and add columns for additional node attributes)
<b>[-w]</b>	nodestate=drained	display only jobs associated with the specified constraint: nodestate (See <a href="#">DISPLAYFLAGS</a> for more information.)

## Output

This command presents detailed node information in whitespace-delineated fields.

The output of this command can be extensive and the values for a number of fields may be truncated. If truncated, the `-v` flag can be used to display full field content.

Column	Format
<b>Name</b>	<NODE NAME>
<b>State</b>	<NODE STATE>
<b>Procs</b>	<AVAILABLE PROCS>:<CONFIGURED PROCS>
<b>Memory</b>	<AVAILABLE MEMORY>:<CONFIGURED MEMORY>
<b>Disk</b>	<AVAILABLE DISK>:<CONFIGURED DISK>
<b>Swap</b>	<AVAILABLE SWAP>:<CONFIGURED SWAP>
<b>Speed</b>	<RELATIVE MACHINE SPEED>

Column	Format
<b>Opsys</b>	<NODE OPERATING SYSTEM>
<b>Arch</b>	<NODE HARDWARE ARCHITECTURE>
<b>Par</b>	<PARTITION NODE IS ASSIGNED TO>
<b>Load</b>	<CURRENT 1 MINUTE BSD LOAD>
<b>Rsv</b>	<NUMBER OF RESERVATIONS ON NODE>
<b>Classes</b>	<CLASS NAME>
<b>Network</b>	<NETWORK NAME>...
<b>Features</b>	<NODE FEATURE>...

## Examples

### Example 3-23:

```

> mdiag -n

compute node summary
Name                State   Procs   Memory   Opsys

opt-001             Busy    0:2    2048:2048   SuSE
opt-002             Busy    0:2    2048:2048   SuSE
opt-003             Busy    0:2    2048:2048   SuSE
opt-004             Busy    0:2    2048:2048   SuSE
opt-005             Busy    0:2    2048:2048   SuSE
opt-006             Busy    0:2    2048:2048   SuSE
WARNING: swap is low on node opt-006
opt-007             Busy    0:2    2048:2048   SuSE
opt-008             Busy    0:2    2048:2048   SuSE
opt-009             Busy    0:2    2048:2048   SuSE
opt-010             Busy    0:2    2048:2048   SuSE
opt-011             Busy    0:2    2048:2048   SuSE
opt-012             Busy    0:2    2048:2048   SuSE
opt-013             Busy    0:2    2048:2048   SuSE
opt-014             Busy    0:2    2048:2048   SuSE
opt-015             Busy    0:2    2048:2048   SuSE
opt-016             Busy    0:2    2048:2048   SuSE
x86-001             Busy    0:1    512:512     Redhat
x86-002             Busy    0:1    512:512     Redhat
x86-003             Busy    0:1    512:512     Redhat
x86-004             Busy    0:1    512:512     Redhat
x86-005             Idle    1:1    512:512     Redhat
x86-006             Idle    1:1    512:512     Redhat
x86-007             Idle    1:1    512:512     Redhat
x86-008             Busy    0:1    512:512     Redhat
x86-009             Down    1:1    512:512     Redhat
x86-010             Busy    0:1    512:512     Redhat
x86-011             Busy    0:1    512:512     Redhat
x86-012             Busy    0:1    512:512     Redhat
x86-013             Busy    0:1    512:512     Redhat
x86-014             Busy    0:1    512:512     Redhat
x86-015             Busy    0:1    512:512     Redhat
x86-016             Busy    0:1    512:512     Redhat
P690-001            Busy    0:1    16384:16384  AIX
P690-002            Busy    0:1    16384:16384  AIX
P690-003            Busy    0:1    16384:16384  AIX
P690-004            Busy    0:1    16384:16384  AIX
P690-005            Busy    0:1    16384:16384  AIX
P690-006            Busy    0:1    16384:16384  AIX
P690-007            Idle    1:1    16384:16384  AIX
P690-008            Idle    1:1    16384:16384  AIX
WARNING: node P690-008 is missing ethernet adapter
P690-009            Busy    0:1    16384:16384  AIX
P690-010            Busy    0:1    16384:16384  AIX
P690-011            Busy    0:1    16384:16384  AIX
P690-012            Busy    0:1    16384:16384  AIX
P690-013            Busy    0:1    16384:16384  AIX
P690-014            Busy    0:1    16384:16384  AIX
P690-015            Busy    0:1    16384:16384  AIX
P690-016            Busy    0:1    16384:16384  AIX
-----
---                6:64    745472:745472  -----

Total Nodes: 36  (Active: 30  Idle: 5  Down: 1)

```

**i** Warning messages are interspersed with the node configuration information with all warnings preceded by the keyword **WARNING**.

## XML Output

If XML output is requested (via the `--format=xml` argument), XML based node information will be written to STDOUT in the following format:

```
mdiag -n --format=xml
<Data>
  <node> <ATTR>="<VAL>" ... </node>
  ...
</Data>
```

## XML Attributes

Name	Description
<b>AGRES</b>	Available generic resources
<b>ALLOCRES</b>	Special allocated resources (like vlans)
<b>ARCH</b>	The node's processor architecture.
<b>AVLCLASS</b>	Classes available on the node.
<b>AVLETIME</b>	Time when the node will no longer be available (used in Utility centers)
<b>AVLSTIME</b>	Time when the node will be available (used in Utility centers)
<b>CFGCLASS</b>	Classes configured on the node
<b>ENABLEPROFILING</b>	If true, a node's state and usage is tracked over time.
<b>FEATURES</b>	A list of comma-separated custom features describing a node.
<b>GEVENT</b>	A user-defined event that allows Moab to perform some action.
<b>GMETRIC</b>	A list of comma-separated consumable resources associated with a node.
<b>GRES</b>	generic resources on the node

Name	Description
<b>HOPCOUNT</b>	How many hops the node took to reach this Moab (used in hierarchical grids)
<b>ISDELETED</b>	Node has been deleted
<b>ISDYNAMIC</b>	Node is dynamic (used in Utility centers)
<b>JOBLIST</b>	The list of jobs currently running on a node.
<b>LOAD</b>	Current load as reported by the resource manager
<b>LOADWEIGHT</b>	Load weight used when calculating node priority
<b>MAXJOB</b>	See <a href="#">Node Policies</a> for details.
<b>MAXJOBPERUSER</b>	See <a href="#">Node Policies</a> for details.
<b>MAXLOAD</b>	See <a href="#">Node Policies</a> for details.
<b>MAXPROC</b>	See <a href="#">Node Policies</a> for details.
<b>MAXPROCPERUSER</b>	See <a href="#">Node Policies</a> for details.
<b>NETWORK</b>	The ability to specify which networks are available to a given node is limited to only a few resource managers. Using the <b>NETWORK</b> attribute, administrators can establish this node to network connection directly through the scheduler. The <a href="#">NODECFG</a> parameter allows this list to be specified in a comma-delimited list.
<b>NODEID</b>	The unique identifier for a node.
<b>NODESTATE</b>	The state of a node.
<b>OS</b>	A node's operating system.
<b>OSLIST</b>	Operating systems the node can run
<b>OSMODACTION</b>	URL for changing the operating system
<b>OWNER</b>	Credential type and name of owner

Name	Description
<b>PARTITION</b>	The partition a node belongs to. See <a href="#">Node Location</a> for details.
<b>POWER</b>	The state of the node's power. Either ON or OFF.
<b>PRIORITY</b>	The fixed node priority relative to other nodes.
<b>PROCSPEED</b>	A node's processor speed information specified in MHz.
<b>RACK</b>	The rack associated with a node's physical location.
<b>RADISK</b>	The total available disk on a node.
<b>RAMEM</b>	The total available memory available on a node.
<b>RAPROC</b>	The total number of processors available on a node.
<b>RASWAP</b>	The total available swap on a node.
<b>RCMEM</b>	The total configured memory on a node.
<b>RCPROC</b>	The total configured processors on a node.
<b>RCSWAP</b>	The total configured swap on a node.
<b>RESCOUNT</b>	Number of reservations on the node
<b>RSVLIST</b>	List of reservations on the node
<b>RESOURCES</b>	Deprecated (use GRES)
<b>RMAccessLIST</b>	A comma-separated list of resource managers who have access to a node.
<b>SIZE</b>	The number of slots or size units consumed by the node.
<b>SLOT</b>	The first slot in the rack associated with the node's physical location.
<b>SPEED</b>	A node's relative speed.
<b>SPEEDWEIGHT</b>	speed weight used to calculate node's priority

Name	Description
<b>STATACTIVETIME</b>	Time node was active
<b>STATMODIFYTIME</b>	Time node's state was modified
<b>STATTOTALTIME</b>	Time node has been monitored
<b>STATUPTIME</b>	Time node has been up
<b>TASKCOUNT</b>	The number of tasks on a node.

Related topics

- [checknode](#)

**mdiag -t**

## Synopsis

```
mdiag -t [-v] [-v] [partitionid]
```

## Overview

The `mdiag -t` command is used to present configuration, usage, health, and diagnostic information about partitions maintained by Moab. The information presented includes partition name, limits, configured and available resources, allocation weights and policies.

## Examples

*Example 3-24: Standard partition diagnostics*

```
> mdiag -t
Partition Status
...
```

**mdiag -p**

## Synopsis

```
mdiag -p [-t partition] [-v]
```

## Overview

The `mdiag -p` command is used to display at a glance information about the job priority configuration and its effects on the current eligible jobs. The information presented by this command includes priority

weights, priority components, and the percentage contribution of each component to the total job priority.

The command hides information about priority components which have been deactivated (i.e. by setting the corresponding component priority weight to 0). For each displayed priority component, this command gives a small amount of context sensitive information. The following table documents this information. In all cases, the output is of the form `<PERCENT>(<CONTEXT INFO>)` where `<PERCENT>` is the percentage contribution of the associated priority component to the job's total priority.

**i** By default, this command only shows information for jobs which are eligible for immediate execution. Jobs which violate soft or hard policies, or have holds, job dependencies, or other job constraints in place will not be displayed. If priority information is needed for any of these jobs, use the `-v` flag or the `checkjob` command.

## Format

Flag	Name	Format	Default	Description	Example
<code>-v</code>	VERBOSE	---	---	Display verbose priority information. If specified, display priority breakdown information for blocked, eligible, and active jobs. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> By default, only information for eligible jobs is displayed. To view blocked jobs in addition to eligible, run <code>mdiag -p -v -v</code>.</p> </div>	<pre>&gt; mdiag -p -v</pre> <div style="border: 1px dashed black; padding: 5px; margin-top: 10px; text-align: center;"> <i>Display priority summary information for eligible and active jobs</i> </div>

## Output

Priority Component	Format	Description
<b>Target</b>	<code>&lt;PERCENT&gt;()</code>	
<b>QOS</b>	<code>&lt;PERCENT&gt;(&lt;QOS&gt;:&lt;QOSPRI&gt;)</code>	QOS — QOS associated with job QOSPRI — Priority assigned to the QOS

Priority Component	Format	Description
<b>FairShare</b>	<PERCENT> ( <USR>:<GRP>:<ACC>:<QOS>:<CLS>)	<i>USR</i> — user fs usage - user fs target <i>GRP</i> — group fs usage - group fs target <i>ACC</i> — account fs usage - account fs target <i>QOS</i> — QOS fs usage - QOS fs target <i>CLS</i> — class fs usage - class fs target
<b>Service</b>	<PERCENT>(<QT>:<XF>:<Byp>)	<i>QTime</i> — job queue time which is applicable towards priority (in minutes) <i>XF</i> — current theoretical minimum XFactor is job were to start immediately <i>Byp</i> — number of times job was bypassed by lower priority jobs via backfill
<b>Resource</b>	<PERCENT> (<NDE>:<PE>:<PRC>:<MEM>)	<i>NDE</i> — nodes requested by job <i>PE</i> — Processor Equivalents as calculated by all resources requested by job <i>PRC</i> — processors requested by job <i>MEM</i> — real memory requested by job

## Examples

### Example 3-25: `mdiag -p`

```
diagnosing job priority information (partition: ALL)

Job          PRIORITY*  Cred(  QOS)  FS(Accnt)  Serv(QTime)
Weights -----  1(   1)    1(   1)    1(   1)
13678        1321*     7.6(100.0) 0.2(  2.7) 92.2(1218.)
13698        235*     42.6(100.0) 1.1(  2.7) 56.3(132.3)
13019        8699     0.6( 50.0) 0.3( 25.4) 99.1(8674.)
13030        8699     0.6( 50.0) 0.3( 25.4) 99.1(8674.)
13099        8537     0.6( 50.0) 0.3( 25.4) 99.1(8512.)
13141        8438     0.6( 50.0) 0.2( 17.6) 99.2(8370.)
13146        8428     0.6( 50.0) 0.2( 17.6) 99.2(8360.)
13153        8360     0.0(  1.0) 0.1( 11.6) 99.8(8347.)
13177        8216     0.0(  1.0) 0.1( 11.6) 99.8(8203.)
13203        8127     0.6( 50.0) 0.3( 25.4) 99.1(8102.)
13211        8098     0.0(  1.0) 0.1( 11.6) 99.8(8085.)
...
13703         137     36.6( 50.0) 12.8( 17.6) 50.6( 69.2)
13702         79      1.3(  1.0)  5.7(  4.5) 93.0( 73.4)

Percent Contribution -----  0.9( 0.9)  0.4( 0.4) 98.7( 98.7)

* indicates system prio set on job
```

The `mdiag -p` command only displays information for priority components actually utilized. In the above example, QoS, Account Fairshare, and QueueTime components are utilized in determining a job's priority. Other components, such as Service Targets, and Bypass are not used and thus are not displayed. (See the [Priority Overview](#) for more information) The output consists of a header, a job by job analysis of jobs, and a summary section.

The header provides column labeling and provides configured priority component and subcomponent weights. In the above example, `QOSWEIGHT` is set to 1000 and `FSWEIGHT` is set to 100. When configuring fairshare, a site also has the option of weighting the individual components of a job's overall fairshare, including its user, group, and account fairshare components. In this output, the QoS and account fairshare weights are set to 1.

The job by job analysis displays a job's total priority and the percentage contribution to that priority of each of the priority components. In this example, job 13019 has a total priority of 8699. Both QoS and Fairshare contribute to the job's total priority although these factors are quite small, contributing 0.6% and 0.3% respectively with the fairshare factor being contributed by an account fairshare target. For this job, the dominant factor is the `service` subcomponent `qtime` which is contributing 99.1% of the total priority since the job has been in the queue for approximately 8600 minutes.

At the end of the job by job description, a Totals line is displayed which documents the average percentage contributions of each priority component to the current idle jobs. In this example, the QoS, Fairshare, and Service components contributed an average of 0.9%, 0.4%, and 98.7% to the jobs' total priorities.

### Related topics

- [Job Priority Overview](#)
- [Moab Cluster Manager - Priority Manager](#)

### `mdiag -q`

## Synopsis

`mdiag -q [qosid]`

## Overview

The `mdiag -q` command is used to present information about each QOS maintained by Moab. The information presented includes QOS name, membership, scheduling priority, weights and flags.

## Examples

### Example 3-26: Standard QOS Diagnostics

```
> mdiag -q
QOS Status
System QOS Settings:  QList: DEFAULT (Def: DEFAULT)  Flags: 0
Name                  * Priority QTWeight QTTarget XFWeight XFTarget   QFlags
JobFlags Limits
DEFAULT              1         1         3         1         5.00  PREEMPTEE
[NONE] [NONE]
  Accounts:  it research
  Classes:  batch
[ALL]                0         0         0         0         0.00  [NONE]
[NONE] [NONE]
high                 1000      1         2         1        10.00  PREEMPTOR
[NONE] [NONE]
  Accounts:  engineering it development research
  Classes:  fast
urgent              10000     1         1         1         7.00  PREEMPTOR
[NONE] [NONE]
  Accounts:  engineering it development
low                 100       1         5         1         1.00  PREEMPTEE
[NONE] [NONE]
  Accounts:  engineering marketing it development research
  Classes:  long bigmem
```

### **mdiag -r**

## Synopsis

```
mdiag -r [reservationid] [-v] [-w type=<type>]
```

## Overview

The `mdiag -r` command allows administrators to look at detailed reservation information. It provides the name, type, partition, starttime and endtime, proc and node counts, as well as actual utilization figures. It also provides detailed information about which resources are being used, how many nodes, how much memory, swap, and processors are being associated with each task. Administrators can also view the Access Control Lists for each reservation as well as any flags that may be active in the reservation. The command gathers information from the Moab cache which prevents it from waiting for the scheduler, but the `--blocking` option can be used to bypass the cache and allow waiting for the scheduler.

The `-w` flag filters the output according to the type of reservation. The allowable reservation types are Job, and User.

## Examples

### Example 3-27:

```

> mdiag -r
Diagnosing Reservations
RsvID          Type Par   StartTime      EndTime        Duration Node Task
Proc
-----
-
engineer.0.1   User  A    -6:29:00      INFINITY      INFINITY      0    0
7
  Flags: STANDINGRSV IGNSTATE OWNERPREEMPT
  ACL:   CLASS==batch+::=long+::=fast+::=bigmem+ QOS==low-::=high+ JATTR==PREEMPTEE+
  CL:    RSV==engineer.0.1
  Task Resources: PROCS: [ALL]
  Attributes (HostExp='fr10n01 fr10n03 fr10n05 fr10n07 fr10n09 fr10n11 fr10n13
fr10n15')
  Active PH: 43.77/45.44 (96.31%)
  SRAttributes (TaskCount: 0 StartTime: 00:00:00 EndTime: 1:00:00:00 Days: ALL)
research.0.2   User  A    -6:29:00      INFINITY      INFINITY      0    0
8
  Flags: STANDINGRSV IGNSTATE OWNERPREEMPT
  ACL:   CLASS==batch+::=long+::=fast+::=bigmem+ QOS==high+::=low- JATTR==PREEMPTEE+
  CL:    RSV==research.0.2
  Task Resources: PROCS: [ALL]
  Attributes (HostExp='fr3n01 fr3n03 fr3n05 fr3n07 fr3n07 fr3n09 fr3n11 fr3n13
fr3n15')
  Active PH: 51.60/51.93 (99.36%)
  SRAttributes (TaskCount: 0 StartTime: 00:00:00 EndTime: 1:00:00:00 Days: ALL)
fast.0.3       User  A     00:14:05      5:14:05       5:00:00      0    0
16
  Flags: STANDINGRSV IGNSTATE OWNERPREEMPT
  ACL:   CLASS==fast+ QOS==high+::=low+::=urgent+::=DEFAULT+ JATTR==PREEMPTEE+
  CL:    RSV==fast.0.3
  Task Resources: PROCS: [ALL]
  Attributes (HostExp='fr12n01 fr12n02 fr12n03 fr12n04 fr12n05 fr12n06 fr12n07
fr12n08 fr12n09 fr12n10 fr12n11 fr12n12 fr12n13 fr12n14 fr12n15 fr12n16')
  SRAttributes (TaskCount: 0 StartTime: 00:00:00 EndTime: 5:00:00 Days:
Mon,Tue,Wed,Thu,Fri)
fast.1.4       User  A     1:00:14:05    1:05:14:05    5:00:00      0    0
16
  Flags: STANDINGRSV IGNSTATE OWNERPREEMPT
  ACL:   CLASS==fast+ QOS==high+::=low+::=urgent+::=DEFAULT+ JATTR==PREEMPTEE+
  CL:    RSV==fast.1.4
  Task Resources: PROCS: [ALL]
  Attributes (HostExp='fr12n01 fr12n02 fr12n03 fr12n04 fr12n05 fr12n06 fr12n07
fr12n08 fr12n09 fr12n10 fr12n11 fr12n12 fr12n13 fr12n14 fr12n15 fr12n16')
  SRAttributes (TaskCount: 0 StartTime: 00:00:00 EndTime: 5:00:00 Days:
Mon,Tue,Wed,Thu,Fri)
job2411        Job   A    -00:01:00     00:06:30      Each tile contains a
summary information about the service it represents, including the following:
  ACL:   JOB==job2411=
  CL:    JOB==job2411 USER==jimf GROUP==newyork ACCT==it CLASS==bigmem QOS==low
JATTR==PREEMPTEE DURATION==00:07:30 PROC==6 PS==2700
job1292        Job   A     00:00:00     00:07:30      00:07:30      0    0
4
  ACL:   JOB==job1292=
  CL:    JOB==job1292 USER==jimf GROUP==newyork ACCT==it CLASS==batch QOS==DEFAULT
JATTR==PREEMPTEE DURATION==00:07:30 PROC==4 PS==1800

```

*Example 3-28:*

With the `-v` option, a nodes line is included for each reservation and shows how many nodes are in the reservation as well as how many tasks are on each node.

```

> mdiag -r -v
Diagnosing Reservations
RsvID          Type Par   StartTime      EndTime        Duration Node Task
Proc
-----
-
Moab.6          Job  B    -00:01:05     00:00:35      00:01:40     1   1
1
  Flags: ISACTIVE
  ACL:   JOB==Moab.6=
  CL:    JOB==Moab.6 USER==tuser1 GROUP==tgroup1 CLASS==fast QOS==starter
  JPRIORITY<=0 DURATION==00:01:40 PROC==1 PS==100
  SubType: JobReservation
  Nodes='node002:1'
  Rsv-Group: Moab.6

Moab.4          Job  B    -00:01:05     00:00:35      00:01:40     1   1
1
  Flags: ISACTIVE
  ACL:   JOB==Moab.4=
  CL:    JOB==Moab.4 USER==tuser1 GROUP==tgroup1 CLASS==batch QOS==starter
  JPRIORITY<=0 DURATION==00:01:40 PROC==1 PS==100
  SubType: JobReservation
  Nodes='node002:1'
  Rsv-Group: Moab.4

Moab.5          Job  A    -00:01:05     00:00:35      00:01:40     3   3
6
  Flags: ISACTIVE
  ACL:   JOB==Moab.5=
  CL:    JOB==Moab.5 USER==tuser1 GROUP==tgroup1 ACCT==marketing CLASS==long
  QOS==low JPRIORITY<=0 DURATION==00:01:40 PROC==6 PS==600
  Task Resources: PROCS: [ALL]
  SubType: JobReservation
  Nodes='node008:1,node007:1,node006:1'
  Rsv-Group: Moab.5

Moab.7          Job  A    -00:01:04     00:00:36      00:01:40     1   1
1
  Flags: ISACTIVE
  ACL:   JOB==Moab.7=
  CL:    JOB==Moab.7 USER==tuser1 GROUP==tgroup1 CLASS==bigmen QOS==starter
  JPRIORITY<=0 DURATION==00:01:40 PROC==1 PS==100
  SubType: JobReservation
  Nodes='node005:1'
  Rsv-Group: Moab.7

Moab.2          Job  A    -00:01:07     3:58:53       4:00:00     1   2
2
  Flags: ISACTIVE
  ACL:   JOB==Moab.2=
  CL:    JOB==Moab.2 USER==tuser1 GROUP==tgroup1 QOS==starter JPRIORITY<=0
  DURATION==4:00:00 PROC==2 PS==28800
  SubType: JobReservation
  Nodes='node009:1'
  Rsv-Group: Moab.2

Moab.8          Job  A     3:58:53      7:58:53       4:00:00     8  16
16
  Flags: PREEMPTTE
  ACL:   JOB==Moab.8=
  CL:    JOB==Moab.8 USER==tuser1 GROUP==tgroup1 ACCT==development CLASS==bigmen

```

```

QOS==starter JPRIORITY<=0 DURATION==4:00:00 PROC==16 PS==230400
  SubType: JobReservation

Nodes='node009:1,node008:1,node007:1,node006:1,node005:1,node004:1,node003:1,node001:
1'
  Attributes (Priority=148)
  Rsv-Group: idle

system.3          User bas   -00:01:08   INFINITY   INFINITY   1   1
2
  Flags: ISCLOSED,ISACTIVE
  ACL:   RSV==system.3=
  CL:   RSV==system.3
  Accounting Creds: User:root
  Task Resources: PROCS: [ALL]
  SubType: Other
  Nodes='node254:1'
  Attributes (HostExp='node254')
  Active PH: 0.00/0.01 (0.00%)
  History: 1322773208:PROCS=2

system.2          User bas   -00:01:08   INFINITY   INFINITY   1   1
2
  Flags: ISCLOSED,ISACTIVE
  ACL:   RSV==system.2=
  CL:   RSV==system.2
  Accounting Creds: User:root
  Task Resources: PROCS: [ALL]
  SubType: Other
  Nodes='node255:1'
  Attributes (HostExp='node255')
  Active PH: 0.00/0.01 (0.00%)
  History: 1322773208:PROCS=2

system.1          User bas   -00:01:08   INFINITY   INFINITY   1   1
2
  Flags: ISCLOSED,ISACTIVE
  ACL:   RSV==system.1=
  CL:   RSV==system.1
  Accounting Creds: User:root
  Task Resources: PROCS: [ALL]
  SubType: Other
  Nodes='node256:1'
  Attributes (HostExp='node256')
  Active PH: 0.00/0.01 (0.00%)
  History: 1322773208:PROCS=2

```

**mdiag -R****Synopsis**

```
mdiag -R [-v] [-V job] [resourcemanagerid]
```

**Overview**

The `mdiag -R` command is used to present information about configured resource managers. The information presented includes name, host, port, state, type, performance statistics and failure

notifications.

## Examples

*Example 3-29:*

```

> $ mdiag -R -v
diagnosing resource managers

RM[internal] State: --- Type: SSS ResourceType: COMPUTE
Max Fail/Iteration: 0
JobCounter: 6
Partition: SHARED
RM Performance: AvgTime=0.00s MaxTime=0.00s (55353 samples)
RM Languages: -
RM Sub-Languages: -

RM[torque] State: Active Type: PBS ResourceType: COMPUTE
Timeout: 30000.00 ms
Version: '4.2.4'
Job Submit URL: exec:///opt/torque-4.2/bin/qsub
Objects Reported: Nodes=1 (12 procs) Jobs=1
Nodes Reported: 1 (N/A)
Flags: executionServer
Partition: torque
Event Management: EPORT=15004 (last event: 00:03:07)
NOTE: SSS protocol enabled
Submit Command: /opt/torque-4.2/bin/qsub
DefaultClass: batch
Total Jobs Started: 1
RM Performance: AvgTime=0.00s MaxTime=35.00s (220097 samples)
RM Languages: PBS
RM Sub-Languages: PBS

RM[torque] Failures:
clusterquery (683 of 55349 failed)
-12days 'cannot connect to PBS server ' (pbs_errno=15033, 'Batch protocol
error')'

NOTE: use 'mrmctl -f messages <RMID>' to clear stats/failures

RM[FLEXlm] State: Active Type: NATIVE ResourceType: LICENSE
Timeout: 30000.00 ms
Cluster Query URL: exec://$TOOLS_DIR/flexlm/license.mon.flexLM.pl
Licenses Reported: 6 types (250 of 282 available)
Partition: SHARED
License Stats: Avg License Avail: 239.01 (978 iterations)
Iteration Summary: Idle: 396.42 Active: 150.92 Busy: -447.34
License biocol 50 of 50 available (Idle: 100.00% Active: 0.00%)
License cloudform 100 of 100 available (Idle: 100.00% Active: 0.00%)
License mathworks 8 of 25 available (Idle: 52.00% Active: 48.00%)
License verity 25 of 25 available (Idle: 100.00% Active: 0.00%)
Event Management: (event interface disabled)
RM Performance: AvgTime=0.00s MaxTime=0.61s (1307618 samples)
clusterquery: AvgTime=0.02s MaxTime=0.61s (9465 samples)
queuequery: AvgTime=0.00s MaxTime=0.00s (1 samples)
rminitialize: AvgTime=0.00s MaxTime=0.00s (1 samples)
getdata: AvgTime=0.17s MaxTime=0.60s (978 samples)
RM Languages: NATIVE
RM Sub-Languages: NATIVE

AM[mam] Type: MAM State: 'Active'
Host: localhost
Port: 7112
Timeout: 15
Thread Pool Size: 2
Charge Policy: DEBITALLWC
Validate Job Submission: TRUE
Create Failure Action: CANCEL,HOLD
Start Failure Action: CANCEL,HOLD

AM[mam] Failures:

```

```
Fri Jun 21 14:32:45 Create          'Failure registering job Create (1) with
accounting manager -- server rejected request with status code 740 - Insufficient
funds: There are no valid allocations to satisfy the quote'
```

## **mdiag -S**

## **Synopsis**

`mdiag -S [-v] [-v]`

## **Overview**

The `mdiag -S` command is used to present information about the status of the scheduler and grid interface.

This command will report on the following aspects of scheduling:

- **General Scheduler Configuration**
  - Reports short and long term scheduler load
  - Reports detected overflows of node, job, reservation, partition, and other scheduler object tables
- **High Availability**
  - Configuration
  - Reports health of HA primary
  - Reports health of HA backup
- **Scheduling Status**
  - Reports if scheduling is paused
  - Reports if scheduling is stopped
- **System Reservation Status**
  - Reports if global system reservation is active
- **Message Profiling/Statistics Status**
- **Moab scheduling activities (only with `mdiag -S -v -v`)**
  - Activity[JobStart]: Time Moab spends telling the RM to start a job and waiting for a response.
  - Activity[RMResourceLoad]: Time Moab spends querying license managers and nodes.
  - Activity[RMWorkloadLoad]: Time Moab spends querying resource managers about jobs (as opposed to nodes)
  - Activity[Schedule]: Time Moab spends prioritizing jobs and scheduling them onto nodes.
  - Activity[UIProcess]: Time Moab spends handling client commands.

## Examples

### Example 3-30:

```
> mdiag -s
Moab Server running on orion-1:43225 (Mode: NORMAL)
Load(5m)  Sched: 12.27%  RMAAction: 1.16%  RMQuery: 75.30%  User: 0.29%  Idle: 10.98%
Load(24h) Sched: 10.14%  RMAAction: 0.93%  RMQuery: 74.02%  User: 0.11%  Idle: 13.80%
HA Fallback Server: orion-2:43225 (Fallback is Ready)
Note: system reservation blocking all nodes
Message: profiling enabled (531 of 600 samples/5:00 interval)
```

### **mdiag -s**

## Synopsis

```
mdiag -s [reservationid] [-v]>
```

## Overview

The `mdiag -s` command allows administrators to look at detailed standing reservation information. It provides the name, type, partition, starttime and endtime, period, task count, host list, and a list of child instances.

## Examples

### Example 3-31:

```
> mdiag -s
standing reservation overview
RsvID          Type      Par      StartTime      EndTime      Duration      Period
-----
TestSR          User      ---      00:00:00      ---          00:00:00      DAY
  Days:          ALL
  Depth:         2
  RsvList:       testSR.1, testSR.2, testSR.3
  HostExp:       'node1, node2, node4, node8'

test2           User      ---      00:00:00      ---          00:00:00      DAY
  Days:          ALL
  TaskCount:     4
  Depth:         1
  RsvList:       test2.4, test2.5
```

### **mdiag -T**

## Synopsis

```
mdiag -T [triggerid] [-v] [--blocking]
```

## Overview

The `mdiag -T` command is used to present information about each Trigger. The information presented includes Name, State, Action, Event Time. The command gathers information from the Moab cache which prevents it from waiting for the scheduler, but the `-blocking` option can be used to bypass the cache and allow waiting for the scheduler.

## Examples

Example 3-32:

```

> mdiag -T
TrigID          Object ID          Event  AType          ActionDate
State
-----
sched_trig.0    sched:Moab        end    exec           -
Blocked
3              node:node010      threshol  exec           -
Blocked
5              job:Moab.7        preempt  exec           -
Blocked
6              job:Moab.8        preempt  exec           -
Blocked
4*             job:Moab.5        start    exec           -00:00:36
Failure
* indicates trigger has completed

```

**Example 3-33:**

```

> mdiag -T -v
TrigID          Object ID          Event  AType          ActionDate
State
-----
sched_trig.0    sched:Moab        end    exec           -
Blocked
  Name:          sched_trig
  Flags:         globaltrig
  BlockUntil:    INFINITY  ActiveTime:  ---
  Action Data:   date
  NOTE: trigger can launch

3              node:node010      threshol  exec           -
Blocked
  Flags:         globaltrig
  BlockUntil:    INFINITY  ActiveTime:  ---
  Threshold:     CPULoad > 3.00 (current value: 0.00)
  Action Data:   date
  NOTE: trigger cannot launch - threshold not satisfied - threshold type not
supported

5              job:Moab.7        preempt  exec           -
Blocked
  Flags:         user,globaltrig
  BlockUntil:    INFINITY  ActiveTime:  ---
  Action Data:   $HOME/tools/preemptnotify.pl $OID $OWNER $HOSTNAME

6              job:Moab.8        preempt  exec           -
Blocked
  Flags:         user,globaltrig
  BlockUntil:    INFINITY  ActiveTime:  ---
  Action Data:   $HOME/tools/preemptnotify.pl $OID $OWNER $HOSTNAME
  NOTE: trigger cannot launch - parent job Moab.8 is in state Idle

4*             job:Moab.5        start    exec   Mon Jan 16 12:33:00
Failure
  Launch Time:   -00:02:17
  Flags:         globaltrig
  Last Execution State: Failure (ExitCode: 0)
  BlockUntil:    00:00:00  ActiveTime:  00:00:00
  Action Data:   $HOME/tools/preemptnotify.pl $OID $OWNER $HOSTNAME
  ALERT: trigger failure detected
  Message:       'exec '/usr/test/moab/tools/preemptnotify.pl' cannot be located or is
not executable'

* indicates trigger has completed

```

**mdiag -u****Synopsis**

```
mdiag -u [userid]
```

**Overview**

The `mdiag -u` command is used to present information about user records maintained by Moab. The information presented includes user name, UID, scheduling priority, default job flags, default QOS level,

List of accessible QOS levels, and list of accessible partitions.

## Examples

Example 3-34:

```

> mdiag -u
evaluating user information
Name          Priority      Flags          QDef          QOSList*      PartitionList
Target Limits
jvella                0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
  ALIST=Engineering
  Message: profiling enabled (597 of 3000 samples/00:15:00 interval)
[NONE]                0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
reynolds              0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
  ALIST=Administration
  Message: profiling enabled (597 of 3000 samples/00:15:00 interval)
mshaw                 0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
  ALIST=Test
  Message: profiling enabled (584 of 3000 samples/00:15:00 interval)
kforbes              0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
  ALIST=Shared
  Message: profiling enabled (597 of 3000 samples/00:15:00 interval)
gastor                0      [NONE]      [NONE]      [NONE]      [NONE]
0.00 [NONE]
  ALIST=Engineering
  Message: profiling enabled (597 of 3000 samples/00:15:00 interval)

```

Note that only users which have jobs which are currently queued or have been queued since Moab was most recently started are listed.

### Related topics

- [showstats](#) command (display user statistics)

## mjobctl

### Synopsis

[mjobctl -c \*jobexp\*](#)

[mjobctl -c -w \*attr=val\*](#)

[mjobctl -C \*jobexp\*](#)

[mjobctl -e \*jobid\*](#)

[mjobctl -F \*jobexp\*](#)

[mjobctl -h](#) [User|System|Batch|Defer|All] [jobexp](#)

[mjobctl -m \*attr{+=|-=}\*val](#)[jobexp](#)

[mjobctl -N](#) [<SIGNO>] [jobexp](#)  
[mjobctl -n](#) <JOBNAME>  
[mjobctl -p](#) <PRIORITY> [jobexp](#)  
[mjobctl -q](#) {diag|starttime|hostlist} [jobexp](#)  
[mjobctl -r](#) [jobexp](#)  
[mjobctl -R](#) [jobexp](#)  
[mjobctl -s](#) [jobexp](#)  
[mjobctl -u](#) [jobexp](#)  
[mjobctl -w](#) attr{+=|-=}val[jobexp](#)  
[mjobctl -x](#) [-w flags=val] [jobexp](#)

## Overview

The `mjobctl` command controls various aspects of jobs. It is used to submit, cancel, execute, and checkpoint jobs. It can also display diagnostic information about each job. The `mjobctl` command enables the Moab administrator to control almost all aspects of job behavior. See [General Job Administration](#) for more details on jobs and their attributes.

## Format

-c - Cancel	
<b>Format</b>	<a href="#">JOBEXP</a>
<b>Description</b>	Cancel a job. <div style="border: 1px solid #4a6980; border-radius: 10px; padding: 10px; margin-top: 10px;">  Use <a href="#">-w</a> (following a <a href="#">-c</a> flag) to specify job cancellation according to given credentials or job attributes. See <a href="#">-c -w</a> for more information.           </div>
<b>Example:</b>	<pre style="border: 1px dashed #ccc; padding: 10px;"> &gt; mjobctl -c job1045  Cancel job job1045.</pre>

-c -w - Cancel Where	
<b>Format</b>	<ATTR>=<VALUE>  where <ATTR>=[ user   account   qos   class   reqreservation(RsvName)   state (JobState)   job-name(JobName, not job ID)]   partition

**-c -w - Cancel Where**

<b>Description</b>	<p>Cancel a job based on a given credential or job attribute.</p> <p>Use <code>-w</code> following a <code>-c</code> flag to specify job cancellation according to credentials or job attributes. (See examples.)</p> <p>See <a href="#">Job States</a> on page 99 for a list of all valid job states.</p> <p>Also, you can cancel jobs from given partitions using <code>-w partition=&lt;PAR1&gt;[&lt;PAR2&gt;...]</code>; however, you must also either use another <code>-w</code> flag to specify a job or use the standard job expression.</p>
<b>Example</b>	<pre>&gt; mjobctl -c -w state=USERHOLD</pre> <p><i> Cancels all jobs that currently have a USERHOLD on them.</i></p> <pre>&gt; mjobctl -c -w user=user1 -w acct=acct1</pre> <p><i> Cancels all jobs assigned to user1 or acct1.</i></p>

**-C - Checkpoint**

<b>Format</b>	<a href="#">JOBEXP</a>
<b>Description</b>	Checkpoint a job. See <a href="#">Checkpoint/Restart Facilities</a> on page 528 for more information.
<b>Example</b>	<pre>&gt; mjobctl -C job1045</pre> <p><i> Checkpoint job job1045.</i></p>

**-e - Rerun**

<b>Format</b>	JOBID
<b>Description</b>	Rerun the completed TORQUE job. This works only for jobs that are completed and show up in TORQUE as completed. This flag does not work with other resource managers.
<b>Example</b>	<pre>&gt; mjobctl -e job1045</pre> <p><i> Rerun job job1045.</i></p>

-F - Force Cancel	
<b>Format</b>	<u>JOBEXP</u>
<b>Description</b>	Forces a job to cancel and ignores previous cancellation attempts.
<b>Example</b>	<pre>&gt; mjobctl -F job1045</pre> <p><i>Force cancel job job1045.</i></p>

-h - Hold	
<b>Format</b>	<p>&lt;HOLDTYPE&gt;&lt;JOBEXP&gt;</p> <p>&lt;HOLDTYPE&gt; = { user   batch   system   defer   ALL }</p>
<b>Default</b>	user
<b>Description</b>	<p>Set or release a job hold</p> <p>See <a href="#">Job Holds on page 525</a> for more information</p>
<b>Example</b>	<pre>&gt; mjobctl -h user job1045</pre> <p><i>Set a user hold on job job1045.</i></p> <pre>&gt; mjobctl -u all job1045</pre> <p><i>Unset all holds on job job1045.</i></p>

-m - Modify	
<b>Format</b>	<p>&lt;ATTR&gt;{ +=   =   -= } &lt;VAL&gt;</p> <p>&lt;ATTR&gt;={ account   arraylimit   awduration   class   cpuclock   deadline   depend   eeduration   env   features   feature   flags   gres   group   hold   hostlist   jobdisk   jobmem   jobname   job-swap   loglevel   messages   minstarttime   nodecount   notificationaddress   partition   priority   queue   qos   reqreservation   rmxstring   reqattr   reqawduration   sysprio   trig   trigvar   userprio   var   wclimit}</p>

**-m - Modify****Description**

Modify a specific job attribute.

**i** If an `mjobctl -m` attribute can affect how a job starts, then it generally cannot affect a job that is already running. For example, it is not feasible to change the **hostlist** of a job that is already running.

The `userprio` attribute allows you to specify user priority. For job priority, use the `'-p'` flag.

Modification of the job dependency is also communicated to the resource manager in the case of SLURM and PBS/TORQUE.

Adding `--flags=warnifcompleted` causes a warning message to print when a job completes.

To define values for `awduration`, `eeduration`, `minstarttime` (Note that the `minstarttime` attribute performs the same function as [msub -a.](#)), `reqawduration`, and `wclimit`, use the [time spec](#) format.

A non-active job's partition list can be modified by adding or subtracting partitions. Note, though, that when adding or subtracting multiple partitions, each partition must have its own `-m partition{+= | = | -=}name` on the command line. (See example for adding multiple partitions.)

To modify a job's generic resources, use the following format: `gres{ += | = | -= } <gresName>[:<count>].` `<gresName>` is a single resource, not a list. `<count>` is an integer that, if not specified, is assumed to be 1. Modifying a job's generic resources causes Moab to append the new gres (`+=`), subtract the specified gres (`-=`), or clear out all existing generic resources attached to the job and override them with the newly-specified one (`=`).

**-m - Modify****Example**

```
> mjobctl -m reqawduration+=600 1664
```

*Add 10 minutes to the job walltime.*

```
> mjobctl -m eeduration=-1 1664
```

*Reset job's effective queue time, to when the job was submitted.*

```
> mjobctl -m var=Flag1=TRUE 1664
```

*Set the job variable `Flag1` to `TRUE`.*

```
> mjobctl -m notificationaddress="name@server.com"
```

*Sets the notification e-mail address associated with a job to `name@server.com`.*

```
> mjobctl -m partition+=p3 -m partition+=p4 Moab.5
```

*Adds multiple partitions (`p3` and `p4`) to job `Moab.5`.*

```
> mjobctl -m arraylimit=10 sim.25
```

*Changes the concurrently running sub-job limit to 10 for array `sim.25`.*

```
> mjobctl -m gres=matlab:1 job0201
```

*Overrides all generic resources applied to job `job0201` and replaces them with 1 `matlab`.*

```
> mjobctl -m userprio-=100 Moab.4
```

*Reduces the user priority of `Moab.4` by 100.*

**-N - Notify****Format**

```
[signal=]<SIGID>JOBEXP
```

**Description**

Send a signal to all jobs matching the job expression.

**-N - Notify**

<b>Example</b>	<pre>&gt; mjobctl -N INT 1664</pre> <p><i>Send an interrupt signal to job 1664.</i></p> <pre>&gt; mjobctl -N 47 1664</pre> <p><i>Send signal 47 to job 1664.</i></p>
----------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>-n - Name</b>	
<b>Format</b>	
<b>Description</b>	Select jobs by job name.
<b>Example</b>	

<b>-p - Priority</b>	
<b>Format</b>	[+ = -]=<VAL><JOBID> [--flags=relative]
<b>Description</b>	Modify a job's system priority.

**-p - Priority****Example**

Priority is the job priority plus the system priority. Each format affects the job and system priorities differently. Using the format `<VAL><JOBID>` or `+<VAL><JOBID>` will set the system priority to the maximum system priority plus the specified value. Using `+=<VAL><JOBID>` or `<VAL><JOBID> --flags=relative` will relatively increase the job's priority and set the system priority. Using the format `--<VAL> <JOBID>` sets the system priority to 0, and does not change priority based on `<VAL>` (it will not decrease priority by that number).

For the following example, `job1045` has a priority of 10, which is composed of a job priority of 10 and a system priority of 0.

```
> mjobctl -p +1000 job1045
```

*The system priority changes to the max system priority plus 1000 points, ensuring that this job will be higher priority than all normal jobs. In this case, the job priority of 10 is not added, so the priority of `job1045` is now 1000001000.*

```
> mjobctl -p -=1 job1045
```

*The system priority of `job1045` resets to 0. The job priority is still 10, so the overall priority becomes 10.*

```
> mjobctl -p 3 job1045 --flags=relative
```

*Adds 3 points to the relative system priority. The priority for `job1045` changes from 10 to 13.*

**-q - Query**

**Format** [ diag( ALL) | hostlist | starttime | template ] <JOBEXP>

**Description** Query a job.

## -q - Query

### Example

```
> mjobctl -q diag job1045
```

*Query job job1045.*

```
> mjobctl -q diag ALL --format=xml
```

*Query all jobs and return the output in machine-readable XML.*

```
> mjobctl -q starttime job1045
```

*Query starttime of job job1045.*

```
> mjobctl -q template <job>
```

*Query job templates. If the <job> is set to ALL or empty, it will return information for all job templates.*

```
> mjobctl -q wiki <jobName>
```

*Query a job with the output displayed in a WIKI string. The job's name may be replaced with ALL.*



*--flags=completed will only work with the diag option.*

## -r - Resume

### Format

[JOBEXP](#)

### Description

Resume a job.

### Example

```
> mjobctl -r
job1045
```

*Resume
job
job1045.*

## -R - Requeue

### Format

[JOBEXP](#)

**-R - Requeue**

**Description** Requeue a job. Adding `--flags=unmigrate` causes Moab to pull a grid job back to the central scheduler for further evaluation on all valid partitions.

**Example**

```
> mjobctl -R job1045
```

*Requeue job job1045.*

**-s - Suspend**

**Format** [JOBEXP](#)

**Description** Suspend a job. For more information, see [Suspend/Resume Handling](#).

**Example**

```
> mjobctl -s job1045
```

*Suspend job job1045.*

**-u - Unhold**

**Format** [`<TYPE>` [`<TYPE>`]] [JOBEXP](#)  
`<TYPE>` = [ user | system | batch | defer | ALL ]

**Default** ALL

**Description** Release a hold on a job  
 See [Job Holds on page 525](#) for more information.

**Example**

```
> mjobctl -u user,system scrib.1045
```

*Release user and system holds on job scrib.1045.*

**-w - Where**

**Format** [CompletionTime | StartTime][<= | = | >=]<EPOCH\_TIME>

-w - Where	
<b>Description</b>	Add a where constraint clause to the current command. As it pertains to <code>CompletionTime</code>   <code>StartTime</code> , the where constraint only works for completed jobs. <code>CompletionTime</code> filters according to the completed jobs' completion times; <code>StartTime</code> filters according to the completed jobs' start times.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mjobctl -q diag ALL --flags=COMPLETED --format=xml -w CompletionTime&gt;=1246428000 -w CompletionTime&lt;=1254376800</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 10px;"><i>Prints all completed jobs still in memory that completed between July 1, 2009 and October 1, 2009.</i></p>

-x - Execute	
<b>Format</b>	<a href="#">JOBEXP</a>
<b>Description</b>	Execute a job. The <code>-w</code> option allows flags to be set for the job. Allowable flags are, <code>ignorepolicies</code> , <code>ignorenodestate</code> , and <code>ignorersv</code> .
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mjobctl -x job1045</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Execute job job1045.</i></p> <pre style="border: 1px dashed gray; padding: 5px; margin-top: 10px;">&gt; mjobctl -x -w flags=ignorepolicies job1046</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Execute job job1046 and ignore policies, such as MaxJobPerUser.</i></p>

## Parameters

JOB EXPRESSION	
<b>Format</b>	<STRING>

## JOB EXPRESSION

**Description**

The name of a job or a regular expression for several jobs. The flags that support job expressions can use node expression syntax as described in [Node Selection](#). Using `x:` indicates the following string is to be interpreted as a regular expression, and using `r:` indicates the following string is to be interpreted as a range. Job expressions do not work for array sub-jobs.

**i** Moab uses regular expressions conforming to the POSIX 1003.2 standard. This standard is somewhat different than the regular expressions commonly used for filename matching in Unix environments (see man 7 regex). To interpret a job expression as a regular expression, use `x:` or in the Moab configuration file (`moab.cfg`), set the parameter **USEJOBREGEX** to **TRUE** (and take note of the following caution).

**!** If you set **USEJOBREGEX** to **TRUE**, Moab treats all `mjobctl` job expressions as regular expressions regardless of whether wildcards are specified. This should be used with extreme caution since there is high potential for unintended consequences. For example, specifying `canceljob m.1` will not only cancel `m.1`, but also `m.11`, `m.12`, `m.13`, and so on.

**i** In most cases, it is necessary to quote the job expression (for example, `job13[5-9]`) to prevent the shell from intercepting and interpreting the special characters.

**i** The `mjobctl` command accepts a comma delimited list of job expressions. Example usage might be `mjobctl -r job[1-2],job4` or `mjobctl -c job1,job2,job4`.

**Example:**

```
> mjobctl -c "x:80.*"
job '802' cancelled
job '803' cancelled
job '804' cancelled
job '805' cancelled
job '806' cancelled
job '807' cancelled
job '808' cancelled
job '809' cancelled
```

*Cancel all jobs starting with 80.*

```
> mjobctl -m priority+=200 "x:74[3-5]"
job '743' system priority modified
job '744' system priority modified
job '745' system priority modified
```

```
> mjobctl -h x:17.*
# This puts a hold on any job that has a 17 that is followed by an unlimited amount
of any
# character and includes jobs 1701, 17mjk10, and 17DjN_JW-07

> mjobctl -h r:1-17
# This puts a hold on jobs 1 through 17.
```

## XML Output

mjobctl information can be reported as XML as well. This is done with the command `mjobctl -q diag <JOB_ID>`.

### XML Attributes

Name	Description
<b>Account</b>	The account assigned to the job
<b>AllocNodeList</b>	The nodes allocated to the job
<b>Args</b>	The job's executable arguments
<b>AWDuration</b>	The active wall time consumed
<b>BlockReason</b>	The block message index for the reason the job is not eligible
<b>Bypass</b>	Number of times the job has been bypassed by other jobs
<b>Calendar</b>	The job's timeframe constraint calendar
<b>Class</b>	The class assigned to the job
<b>CmdFile</b>	The command file path
<b>CompletionCode</b>	The return code of the job as extracted from the RM
<b>CompletionTime</b>	The time of the job's completion
<b>Cost</b>	The cost of executing the job relative to an allocation manager
<b>CPULimit</b>	The CPU limit for the job
<b>Depend</b>	Any dependencies on the status of other jobs
<b>DRM</b>	The master destination RM
<b>DRMJID</b>	The master destination RM job ID
<b>EEDuration</b>	The duration of time the job has been eligible for scheduling

Name	Description
<b>EFile</b>	The stderr file
<b>Env</b>	The job's environment variables set for execution
<b>EnvOverride</b>	The job's overriding environment variables set for execution
<b>EState</b>	The expected state of the job
<b>EstHistStartTime</b>	The estimated historical start time
<b>EstPrioStartTime</b>	The estimated priority start time
<b>EstRsvStartTime</b>	The estimated reservation start time
<b>ExchList</b>	The excluded host list
<b>Flags</b>	Command delimited list of Moab flags on the job
<b>GAttr</b>	The requested generic attributes
<b>GJID</b>	The global job ID
<b>Group</b>	The group assigned to the job
<b>Hold</b>	The hold list
<b>Holdtime</b>	The time the job was put on hold
<b>HopCount</b>	The hop count between the job's peers
<b>HostList</b>	The requested host list
<b>IFlags</b>	The internal flags for the job
<b>IsInteractive</b>	If set, the job is interactive
<b>IsRestartable</b>	If set, the job is restartable
<b>IsSuspendable</b>	If set, the job is suspendable

Name	Description
<b>IWD</b>	The directory where the job is executed
<b>JobID</b>	The job's batch ID.
<b>JobName</b>	The user-specified name for the job
<b>JobGroup</b>	The job ID relative to its group
<b>LogLevel</b>	The individual log level for the job
<b>MasterHost</b>	The specified host to run primary tasks on
<b>Messages</b>	Any messages reported by Moab regarding the job
<b>MinPreemptTime</b>	The minimum amount of time the job must run before being eligible for preemption
<b>Notification</b>	Any events generated to notify the job's user
<b>OFile</b>	The stdout file
<b>OldMessages</b>	Any messages reported by Moab in the old message style regarding the job
<b>OWCLimit</b>	The original wallclock limit
<b>PAL</b>	The partition access list relative to the job
<b>QueueStatus</b>	The job's queue status as generated this iteration
<b>QoS</b>	The QoS assigned to the job
<b>QoSReq</b>	The requested QoS for the job
<b>ReqAWDuration</b>	The requested active walltime duration
<b>ReqCMaxTime</b>	The requested latest allowed completion time
<b>ReqMem</b>	The total memory requested/dedicated to the job
<b>ReqNodes</b>	The number of requested nodes for the job

Name	Description
<b>ReqProcs</b>	The number of requested procs for the job
<b>ReqReservation</b>	The required reservation for the job
<b>ReqRMType</b>	The required RM type
<b>ReqSMinTime</b>	The requested earliest start time
<b>RM</b>	The master source resource manager
<b>RMXString</b>	The resource manager extension string
<b>RsvAccess</b>	The list of reservations accessible by the job
<b>RsvStartTime</b>	The reservation start time
<b>RunPriority</b>	The effective job priority
<b>Shell</b>	The execution shell's output
<b>SID</b>	The job's system ID (parent cluster)
<b>Size</b>	The job's computational size
<b>STotCPU</b>	The average CPU load tracked across all nodes
<b>SMaxCPU</b>	The max CPU load tracked across all nodes
<b>STotMem</b>	The average memory usage tracked across all nodes
<b>SMaxMem</b>	The max memory usage tracked across all nodes
<b>SRMJID</b>	The source RM's ID for the job
<b>StartCount</b>	The number of the times the job has tried to start
<b>StartPriority</b>	The effective job priority
<b>StartTime</b>	The most recent time the job started executing

Name	Description
<b>State</b>	The state of the job as reported by Moab
<b>StatMSUtl</b>	The total number of memory seconds utilized
<b>StatPSDed</b>	The total number of processor seconds dedicated to the job
<b>StatPSUtl</b>	The total number of processor seconds utilized by the job
<b>StdErr</b>	The path to the stderr file
<b>StdIn</b>	The path to the stdin file
<b>StdOut</b>	The path to the stdout file
<b>StepID</b>	StepID of the job (used with LoadLeveler systems)
<b>SubmitHost</b>	The host where the job was submitted
<b>SubmitLanguage</b>	The RM language that the submission request was performed
<b>SubmitString</b>	The string containing the entire submission request
<b>SubmissionTime</b>	The time the job was submitted
<b>SuspendDuration</b>	The amount of time the job has been suspended
<b>SysPrio</b>	The admin specified job priority
<b>SysMinTime</b>	The system specified min. start time
<b>TaskMap</b>	The allocation taskmap for the job
<b>TermTime</b>	The time the job was terminated
<b>User</b>	The user assigned to the job
<b>UserPrio</b>	The user specified job priority
<b>UtlMem</b>	The utilized memory of the job

Name	Description
<b>UtilProcs</b>	The number of utilized processors by the job
<b>Variable</b>	
<b>VWCTime</b>	The virtual wallclock limit

## Examples

Example 3-35:

```
> mjobctl -q diag ALL --format=xml
<Data><job AWDuration="346" Class="batch" CmdFile="jobsleep.sh" EEDuration="0"
EState="Running" Flags="RESTARTABLE" Group="test" IWD="/home/test" JobID="11578"
QOS="high"
RMJID="11578.lolo.icluster.org" ReqAWDuration="00:10:00" ReqNodes="1" ReqProcs="1"
StartCount="1"
StartPriority="1" StartTime="1083861225" StatMSUtl="903.570" StatPSDed="364.610"
StatPSUtl="364.610"
State="Running" SubmissionTime="1083861225" SuspendDuration="0" SysPrio="0"
SysSminTime="00:00:00"
User="test"><req AllocNodeList="hana" AllocPartition="access" ReqNodeFeature="[NONE]"
ReqPartition="access"></req></job><job AWDuration="346" Class="batch"
CmdFile="jobsleep.sh"
EEDuration="0" EState="Running" Flags="RESTARTABLE" Group="test" IWD="/home/test"
JobID="11579"
QOS="high" RMJID="11579.lolo.icluster.org" ReqAWDuration="00:10:00" ReqNodes="1"
ReqProcs="1"
StartCount="1" StartPriority="1" StartTime="1083861225" StatMSUtl="602.380"
StatPSDed="364.610"
StatPSUtl="364.610" State="Running" SubmissionTime="1083861225" SuspendDuration="0"
SysPrio="0"
SysSminTime="00:00:00" User="test"><req AllocNodeList="lolo" AllocPartition="access"
ReqNodeFeature="[NONE]" ReqPartition="access"></req></job></Data>
```

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [setspri](#)
- [canceljob](#)
- [runjob](#)

## mnodectl

### Synopsis

```
mnodectl -m attr{=|-=}val nodeexp  
mnodectl -q \[cat|diag|profile|wiki\] nodeexp
```

## Overview

Change specified attributes for a given [node expression](#).

## Access

By default, this command can be run by any Moab Administrator.

## Format

-m - Modify	
<b>Format</b>	<p>&lt;ATTR&gt;{= - = +=}&lt;VAL&gt;</p> <p>Where &lt;ATTR&gt; is one of the following:  <a href="#">FEATURES</a>  <a href="#">GEVENT</a>,  <a href="#">GMETRIC</a>,  <a href="#">MESSAGE</a>,  <a href="#">OS</a>,  <a href="#">POWER</a>,  <a href="#">STATE</a>,  <a href="#">VARIABLE</a></p> <p>and -=, except when used for features, clears the attribute instead of decrementing the attribute's value and = indicates that you are specifying a new value to replace the old one(s), if any.</p> <p>When the -= option is used to modify features, it removes the specified features from the node. The += option, which is only available for features, allows you to append additional features to the current list rather than replacing the current list entirely.</p> <div style="border: 1px solid #4a6984; border-radius: 10px; padding: 10px; background-color: #d9e1f2; margin-top: 10px;"> <p><b>i</b> Changing OS and POWER require a Moab Adaptive Computing Suite license and a provisioning resource manager.</p> </div>
<b>Description</b>	Modify the state or attribute of <a href="#">specified</a> node(s)
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 10px;"> &gt; mnodectl -m features+=fastio,highmem node1 &gt; mnodectl -m gevent=cpufail:'cpu02 has failed w/ec:0317' node1 &gt; mnodectl -m gmetric=temp:131.2 node1 &gt; mnodectl -m message='cpufailure:cpu02 has failed w/ec:0317' node1 &gt; mnodectl -m OS=RHAS30 node1 &gt; mnodectl -m power=off node1 &gt; mnodectl -m state=idle node1 &gt; mnodectl -m variable=IP=10.10.10.100,Location=R1S2 node1                     </pre>

-q - Query	
<b>Format</b>	{cat   diag   profile   wiki}

**-q - Query****Description**

Query node categories or node profile information (see [ENABLEPROFILING](#) for nodes).



The diag and profile options must use --xml.

**Example**

```
> mnodectl -q cat ALL
node categorization stats from Mon Jul 10 00:00:00 to Mon Jul 10 15:30:00
Node: moab
  Categories:
                busy: 96.88%
                idle: 3.12%
Node: maka
  Categories:
                busy: 96.88%
                idle: 3.12%
Node: pau
  Categories:
                busy: 96.88%
                idle: 3.12%
Node: maowu
  Categories:
                busy: 96.88%
                down-hw: 3.12%
Cluster Summary:
                busy: 96.88%
                down-hw: 0.78%
                idle: 2.34%
```

```
> mnodectl -v -q profile
...
```

```
> mnodectl -q wiki <ALL>
GLOBAL STATE=Idle PARTITION=SHARED
n0 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n1 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n2 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n3 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n4 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n5 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n6 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n7 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n8 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
n9 STATE=Idle PARTITION=base APROC=4 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED
```

*Query a node with the output displayed in a WIKI string.*

**Parameters**

FEATURES	
<b>Format</b>	<p>&lt;STRING&gt;</p> <p>One of the following:</p> <ul style="list-style-type: none"> <li>• a comma-delimited list of features</li> <li>• [NONE] (to clear features on the node)</li> </ul>
<b>Description</b>	<p>Sets the features on a node.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> These node features will be overwritten when an RM reports features.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">mnodectl -m features=fastio,highmem node1 mnodectl -m features=[NONE] node1</pre>

GEVENT	
<b>Format</b>	<EVENT>:<MESSAGE>
<b>Description</b>	Creates a generic event on the node to which Moab may respond (see <a href="#">Enabling Generic Events</a> ).
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">mnodectl -m gevent=powerfail:'power has failed' node1</pre>

GMETRIC	
<b>Format</b>	<ATTR>:<VALUE>
<b>Description</b>	<p>Sets the value for a generic metric on the node (see <a href="#">Enabling Generic Metrics</a>).</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> When a gmetric set in Moab conflicts with what the resource manager reports, Moab uses the set gmetric until the next time the resource manager reports a different number.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">mnodectl -m gmetric=temp:120 node1</pre>

MESSAGE	
<b>Format</b>	'<MESSAGE>'

## MESSAGE

**Description** Sets a message to be displayed on the node.

**Example**

```
mnodectl -m message='powerfailure: power has failed'
node1
```

## NODEEXP

**Format** *<STRING>*  
Where *<NODEEXP>* is a node name, regex or ALL

**Description** Identifies one or more nodes.

**Example** node1 — applies only to node1  
fr10n\* - all nodes starting with fr10n  
ALL - all known nodes

## OS

**Format** *<STRING>*

**Description** Operating System (see [Resource Provisioning](#)).

**Example**

```
mnodectl node1 -m OS=RHELAS30
```

## POWER

**Format** {off|on}

POWER	
<b>Description</b>	<p>Set the power state of a node. Action will NOT be taken if the node is already in the specified state.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;"> <p><b>i</b> If you power off a node, a green policy will try to turn it back on. If you want the node to remain powered off, you must associate a reservation with it.</p> </div> <div style="border: 1px solid #ccc; padding: 5px;"> <p><b>i</b> If you request to power off a node that has active work on it, Moab returns a status indicating that the node is busy (with a job or VM) and will not be powered off. You will see one of these messages:</p> <ul style="list-style-type: none"> <li>• Ignoring node <i>&lt;name&gt;</i>: power ON in process (indicates node is currently powering on)</li> <li>• Ignoring node <i>&lt;name&gt;</i>: power OFF in process (indicates node is currently powering off)</li> <li>• Ignoring node <i>&lt;name&gt;</i>: has active VMs running (indicates the node is currently running active VMs)</li> <li>• Ignoring node <i>&lt;name&gt;</i>: has active jobs running (indicates the node is currently running active jobs)</li> </ul> <p>Once you resolve the activity on the node (by preempting or migrating the jobs or VMs, for example), you can attempt to power the node off again.</p> <p>You can use the <code>--flags=force</code> option to cause a force override. However, doing this will power off the node regardless of whether or not its jobs get migrated or preempted (i.e., you run the risk of losing the VMs/jobs entirely). For example:</p> <div style="border: 1px dashed #ccc; padding: 5px; margin-top: 10px;"> <pre>&gt; mnodectl node1 -m power=off --flags=force</pre> </div> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 5px;"> <pre>&gt; mnodectl node1 -m power=off</pre> </div>

STATE	
<b>Format</b>	{drained idle}
<b>Description</b>	Remove (drained) or add (idle) a node from scheduling.
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 5px;"> <pre>mnodectl node1 -m state=drained</pre> </div> <div style="border: 1px dashed #ccc; padding: 5px;"> <p><i>Moab ignores node1 when scheduling.</i></p> </div>

VARIABLE	
<b>Format</b>	<i>&lt;name&gt;</i> [= <i>&lt;value&gt;</i> ], <i>&lt;name&gt;</i> [= <i>&lt;value&gt;</i> ]..

VARIABLE	
<b>Description</b>	Set a list of variables for a node.
<b>Example</b>	<pre>&gt; mnodectl node1 -m variable=IP=10.10.10.100,Location=R1S2</pre>

### Related topics

- [Moab Client Installation](#) — explains how to distribute this command to client nodes
- [mddiag -n](#)
- [showres -n](#)
- [checknode](#)
- [showstats -n](#) — report current and historical node statistics

## moab

### Synopsis

```
moab --about --help --loglevel=<LOGLEVEL> --version [-c <CONFIG_FILE>] [-C] [-d] [-e] [-h] [-P
<PAUSEDURATION>]] [-R <RECYCLEDURATION>] [-s] [-S <STOPITERATION>]] [-v]
```

### Parameters

Parameter	Description
<b>--about</b>	Displays build environment and version information.
<b>--loglevel</b>	Sets the server loglevel to the specified value.
<b>--version</b>	Displays version information.
<b>-c</b>	Configuration file the server should use.
<b>-C</b>	Clears checkpoint files (.moab.ck, .moab.ck.1).
<b>-d</b>	Debug mode (does not background itself).

Parameter	Description
<b>-e</b>	Forces Moab to exit if there are any errors in the configuration file, if it can't connect to the configured database, or if it can't find these directories: <ul style="list-style-type: none"> <li>• statdir</li> <li>• logdir</li> <li>• spooldir</li> <li>• toolsdir</li> </ul>
<b>-P</b>	Starts Moab in a paused state for the duration specified.
<b>-R</b>	Causes Moab to automatically <a href="#">recycle</a> every time the specified duration transpires.
<b>-s</b>	Starts Moab in the state that was most recently checkpointed.
<b>-S</b>	Suspends/stops scheduling at specified iteration (or at startup if no iteration is specified).
<b>-v</b>	Same as <a href="#">--version</a> .

## mrmctl

### Synopsis

```
mrmctl -f [fobject] {rmName|am:[amid]} mrmctl -l [rmid|am:[amid]] mrmctl -m <attr>=<value> [rmid|am:[amid]] mrmctl -p {rmid|am:[amid]} mrmctl -R {AM|ID}{:RMID}}
```

### Overview

mrmctl allows an admin to query, list, modify, and ping the [resource managers](#) and allocation managers in Moab. mrmctl also allows for a queue (often referred to as a class) to be created for a resource manager.

### Access

By default, this command can be run by level 1 and level 2 Moab administrators (see [ADMINCFG](#)).

### Format

<b>-f - Flush Statistics</b>	
<b>Format</b>	[< <i>fobject</i> >] where <i>fobject</i> is optional and one of messages or stats.

-f - Flush Statistics	
<b>Default</b>	If no <i>object</i> is specified, then reported failures and performance data will be flushed. If no resource manager id is specified, the first resource manager will be flushed.
<b>Description</b>	Clears resource manager statistics. If messages is specified, then reported failures, performance data, and messages will be flushed.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mrmctl -f base</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Moab will clear the statistics for RM base.</i></p>

-l - List	
<b>Format</b>	N/A
<b>Default</b>	All RMs and AMs (when no RM/AM is specified)
<b>Description</b>	List Resource and Allocation Manager(s)
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mrmctl -l</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Moab will list all resource and allocation managers.</i></p>

-m - Modify	
<b>Format</b>	N/A
<b>Default</b>	All RMs and AMs (when no RM/AM is specified).
<b>Description</b>	Modify Resource and Allocation Manager(s).
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mrmctl -m state=disabled peer13</pre>

-p - Ping	
<b>Format</b>	N/A

-p - Ping	
<b>Default</b>	First RM configured.
<b>Description</b>	Ping Resource Manager.
<b>Example</b>	<pre>&gt; mrmctl -p base</pre> <div style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"> <p><i>Moab will ping RM base.</i></p> </div>

-R - Reload	
<b>Format</b>	{AM ID}[:RMID]}
<b>Description</b>	Dynamically reloads server information for the <a href="#">identity manager</a> service if ID is specified; if AM is specified, reloads the allocation manager service.
<b>Example</b>	<pre>&gt; mrmctl -R ID</pre> <p>Reloads the identity manager on demand.</p>

**i** Resource manager interfaces can be enabled/disabled using the modify operation to change the resource manager state as in the following example:

```
# disable active resource manager interface
> mrmctl -m state=disabled torque
# restore disabled resource manager interface
> mrmctl -m state=enabled torque
```

Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mdiag -R](#)
- [mdiag -c](#)

mrsvctl

Synopsis

`mrsvctl -c [-a acl] [-b subtype] [-d duration] [-D description] [-e endtime] [-E] [-f features] [-F flags] [-g rsvgroup] [-h hostexp] [-n name] [-o owner] [-p partition] [-P profile] [-R resources] [-s starttime] [-S setvalue] [-t tasks] [-T trigger] [-V variable] [-x joblist]`

```
mrsvctl -C [-g standing_reservationid] {reservationid}
```

```
mrsvctl -l [{reservationid | -i index}]
```

```
mrsvctl -m <duration|endtime|reqtaskcount|starttime>{=|+=|-}<VAL> <hostexp>{+=|-}<VAL> <variable>
{+=KEY=VAL|-KEY_TO_REMOVE} {reservationid | -i index}
```

```
mrsvctl -q {reservationid | -i index} [--blocking]
```

```
mrsvctl -r {reservationid | -i index}
```

## Overview

`mrsvctl` controls the creation, modification, querying, and releasing of reservations.

The timeframe covered by the reservation can be specified on either an absolute or relative basis. Only jobs with credentials listed in the reservation's access control list can utilize the reserved resources. However, these jobs still have the freedom to utilize resources outside of the reservation. The reservation will be assigned a name derived from the ACL specified. If no reservation ACL is specified, the reservation is created as a system reservation and no jobs will be allowed access to the resources during the specified timeframe (valuable for system maintenance, etc.). See the [Reservation Overview](#) for more information.

Reservations can be viewed using the `-q` flag and can be released using the `-r` flag.

**i** By default, reservations are not exclusive and may overlap with other reservations and jobs. Use the '`-E`' flag to adjust this behavior.

## Access

By default, this command can be run by level 1 and level 2 Moab administrators (see [ADMINCFG](#)).

## Format

-a	
Name	ACL

-a	
<b>Format</b>	<p>&lt;TYPE&gt;==&lt;VAL&gt;[, &lt;TYPE&gt;==&lt;VAL&gt;] . . .</p> <p>Where &lt;TYPE&gt; is one of the following:            ACCT,            CLASS,            DURATION,            GROUP,            JATTR,            PROC,            QOS,            RSV, or            USER</p>
<b>Description</b>	<p>List of limitations for access to the reserved resources (See also: <a href="#">ACL Modifiers</a>).</p>
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mrsvctl -c -h node01 -a USER==john+,CLASS==batch-</pre> <p style="text-align: center; margin-top: 5px;"><i>Moab will make a reservation on node01 allowing access to user john and restricting access from class batch when other resources are available to class batch</i></p> </div> <div style="border: 1px dashed gray; padding: 5px;"> <pre>&gt; mrsvctl -m -a USER-=john system.1</pre> <p style="text-align: center; margin-top: 5px;"><i>Moab will remove user john from the system.1 reservation</i></p> </div>

-a

**Notes**

- When you specify multiple credentials, a user must only match one of them in order to access the reservation. To require one or more of the listed limitations for reservation access, each required specification must end with an asterisk (\*). If a user meets the required limitation(s), he or she has access to the reservation (without meeting any that are not marked required).
- There are three different assignment operators that can be used for modifying most credentials in the ACL. The operator == will reassess the list for that particular credential type. The += operator will append to the list for that credential type, and -= will remove from the list. Two other operators are used to specify DURATION and PROC: >= (greater than) and <= (less than).
- To add multiple credentials of the same type with one command, use a colon to separate them. To separate lists of different credential types, use commas. For example, to reassign the user list to consist of users Joe and Bob, and to append the group MyGroup to the groups list on the system.1 reservation, you could use the command `mrsvctl -m -a USER==Joe:Bob, GROUP+=MyGroup system.1`.
- Any of the ACL modifiers may be used. When using them, it is often useful to put single quotes on either side of the assignment command. For example, `mrsvctl -m -a 'USER==&Joe' system.1`.
- Some flags are mutually exclusive. For example, the ! modifier means that the credential is blocked from the reservation and the & modifier means that the credential must run on that reservation. Moab will take the most recently parsed modifier. Modifiers may be placed on either the left or the right of the argument, so `USER==&JOE` and `USER==JOE&` are equivalent. Moab parses each argument starting from right to left on the right side of the argument, then from left to right on the left side. So, if the command was `USER==!Joe&`, Moab would keep the equivalent of `USER==!Joe` because the ! would be the last one parsed.
- You can set a reservation to have a time limit for submitted jobs using DURATION and the \* modifier. For example, `mrsvctl -m -a 'DURATION<=*1:00:00' system.1` would cause the system.1 reservation to not accept any jobs with a walltime greater than one hour. Similarly, you can set a reservation to have a processor limit using PROC and the \* modifier. `mrsvctl -a 'PROC>=2*' system.2` would cause the system.2 reservation to only allow jobs requesting more than 2 procs to run on it.
- You can verify the ACL of a reservation using the `mdiag -r` command.

```
mrsvctl -m -a 'USER==Joe:Bob, GROUP-=BadGroup, ACCT+=GoodAccount, DURATION<=*1:00:00' system.1
```

*Moab will reassign the USER list to be Joe and Bob, will remove BadGroup from the GROUP list, append GoodAccount to the ACCT list, and only allow jobs that have a submitted walltime of an hour or less on the system.1 reservation.*

**-a**

```
mrsvctl -m -a 'USER==Joe,USER==Bob' system.1
```

*Moab will assign the USER list to Joe, and then reassign it again to Bob. The final result will be that the USER list will just be Bob. To add Joe and Bob, use `mrsvctl -m -a USER==Joe:Bob system.1` or `mrsvctl -m -a USER==Joe,USER+=Bob system.1`.*

**-b**

<b>Name</b>	SUBTYPE
<b>Format</b>	One of the <a href="#">node category</a> values or node category shortcuts.
<b>Description</b>	Add subtype to reservation.
<b>Example</b>	<pre>&gt; mrsvctl -c -b SoftwareMaintenance -t ALL</pre> <p><i>Moab will associate the reserved nodes with the <a href="#">node category</a> SoftwareMaintenance.</i></p>

**-c**

<b>Name</b>	CREATE
<b>Format</b>	<ARGUMENTS>
<b>Description</b>	<p>Creates a reservation.</p> <div style="border: 1px solid black; padding: 5px;"> <p> The <code>-x</code> flag, when used with <code>-F ignjobrsv</code>, lets users create reservations but exclude certain nodes from being part of the reservation because they are running specific jobs. The <code>-F</code> flag instructs <code>mrsvctl</code> to still consider nodes with current running jobs.</p> </div>

-c	
<b>Examples</b>	<pre>&gt; mrsvctl -c -t ALL</pre> <p><i>Moab will create a reservation across all system resources.</i></p> <pre>&gt; mrsvctl -c -t 5 -F ignjobrsv -x moab.5,moab.6</pre> <p><i>Moab will create the reservation while assigning the nodes. Nodes running jobs moab5 and moab6 will not be assigned to the reservation.</i></p> <pre>&gt; mrsvctl -c -d INFINITY</pre> <p><i>Moab will create an infinite reservation.</i></p>

-C	
<b>Name</b>	CLEAR
<b>Format</b>	<RSVID>   -g <SRSVID>
<b>Description</b>	Clears any disabled time slots from standing reservations and allows the recreation of disabled reservations
<b>Example</b>	<pre>&gt; mrsvctl -C -g testing</pre> <p><i>Moab will clear any disabled timeslots from the standing reservation testing.</i></p>

-d	
<b>Name</b>	DURATION
<b>Format</b>	[ [ [DD:]HH:]MM:]SS
<b>Default</b>	INFINITY
<b>Description</b>	Duration of the reservation (not needed if ENDTIME is specified)

**-d**

<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -d 5:00:00</pre> <p><i>Moab will create a reservation on node01 lasting 5 hours.</i></p> <pre>mrsvctl -c -d INFINITY</pre> <p><i>Moab will create a reservation with a duration of INFINITY (no endtime).</i></p>
----------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**-D**

<b>Name</b>	DESCRIPTION
<b>Format</b>	<STRING>
<b>Description</b>	Human-readable description of reservation or purpose
<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -d 5:00:00 -D 'system maintenance to test network'</pre> <p><i>Moab will create a reservation on node01 lasting 5 hours.</i></p>

**-e**

<b>Name</b>	ENDTIME
<b>Format</b>	[HH[:MM[:SS]]][_MO[/DD[/YY]]] or +[[DD:]HH:]MM:]SS
<b>Default</b>	INFINITY
<b>Description</b>	Absolute or relative time reservation will end (not required if Duration specified). ENDTIME also supports an epoch timestamp.

-e	
<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -e +3:00:00</pre> <p><i>Moab will create a reservation on node01 ending in 3 hours.</i></p>

-E	
<b>Name</b>	EXCLUSIVE
<b>Description</b>	When specified, Moab will only create a reservation if there are no other reservations (exclusive or otherwise) which would conflict with the time and space constraints of this reservation. If exceptions are desired, the <a href="#">rsvaccesslist</a> attribute can be set or the <a href="#">ignrsv</a> flag can be used.
<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -E</pre> <p><i>Moab will only create a reservation on node01 if no conflicting reservations are found.</i></p> <div style="border: 1px solid #4a7c9c; padding: 5px; margin-top: 10px;"> <p><b>i</b> This flag is only used at the time of reservation creation. Once the reservation is created, Moab allows jobs into the reservation based on the ACL. Also, once the exclusive reservation is created, it is possible that Moab will overlap it with jobs that match the ACL.</p> </div>

-f	
<b>Name</b>	FEATURES
<b>Format</b>	<STRING>[:<STRING>]...
<b>Description</b>	List of node features which must be possessed by the reserved resources. You can use a backslash and pipe to delimit features to indicate that Moab can use one or the other.
<b>Example</b>	<pre>&gt; mrsvctl -c -h node[0-9] -f fast\ slow</pre> <p><i>Moab will create a reservation on nodes matching the expression and which also have either the feature <i>fast</i> or the feature <i>slow</i>.</i></p>

-F	
<b>Name</b>	FLAGS
<b>Format</b>	<flag>[, <flag>]...
<b>Description</b>	Comma-delimited list of flags to set for the reservation (see <a href="#">Managing Reservations</a> for flags).
<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -F ignstate</pre> <p><i>Moab will create a reservation on node01 ignoring any conflicting node states.</i></p>

-g	
<b>Name</b>	RSVGROUP
<b>Format</b>	<STRING>
<b>Description</b>	For a create operation, create a reservation in this reservation group. For list and modify operations, take actions on all reservations in the specified reservation group. The <code>-g</code> option can also be used in conjunction with the <code>-r</code> option to release a reservation associated with a specified group. See <a href="#">Reservation Group</a> for more information.
<b>Example</b>	<pre>&gt; mrsvctl -c -g staff -h 'node0[1-9]'</pre> <p><i>Moab will create a reservation on nodes matching the <a href="#">node expression</a> given and assign it to the reservation group <code>staff</code>.</i></p>

-h	
<b>Name</b>	HOSTLIST
<b>Format</b>	class:<classname>[,<classname>]... or <STRING> or 'r:<nodeNameStart>[<beginRange>-<endRange>] or ALL

-h	
<b>Description</b>	<p><a href="#">Host expression</a> or a class mapping indicating the nodes which the reservation will allocate.</p> <div style="border: 1px solid #d9534f; padding: 5px; margin-top: 10px;">  When you specify a &lt;STRING&gt;, the HOSTLIST attribute is always treated as a regular expression. foo10 will map to foo10, foo101, foo1006, etc. To request an exact host match, the expression can be bounded by the carat and dollar op expression markers as in ^foo10\$.                 </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mrsvctl -c -h 'r:node0[1-9]'</pre> <p><i>Moab will create a reservation on nodes node01, node02, node03, node04, node05, node06, node07, node08, and node09.</i></p> </div> <div style="border: 1px dashed #ccc; padding: 5px;"> <pre>&gt; mrsvctl -c -h class:batch</pre> <p><i>Moab will create a reservation on all nodes which support class/queue batch.</i></p> </div>

-i	
<b>Name</b>	INDEX
<b>Format</b>	<STRING>
<b>Description</b>	Use the reservation index instead of full reservation ID.
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 5px;"> <pre>&gt; mrsvctl -m -i 1 starttime=+5:00</pre> </div> <div style="border: 1px dashed #ccc; padding: 5px;"> <p><i>Moab will create a reservation on nodes matching the expression given.</i></p> </div>

-l	
<b>Name</b>	LIST
<b>Format</b>	<p>&lt;RSV_ID&gt; or ALL</p> <p>RSV_ID can be the name of a reservation or a regular expression.</p>

-l	
<b>Default</b>	ALL
<b>Description</b>	List reservation(s).
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 5px;"> <pre>&gt; mrsvctl -l system*</pre> </div> <div style="border: 1px dashed gray; padding: 5px;"> <p><i>Moab will list all of the reservations whose names start with system.</i></p> </div>

-m															
<b>Name</b>	MODIFY														
<b>Format</b>	<p>&lt;ATTR&gt;=&lt;VAL&gt;[-m &lt;ATTR2&gt;=&lt;VAL2&gt;]...</p> <p>Where &lt;ATTR&gt; is one of the following:</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th colspan="2" style="background-color: #4a7c9c; color: white;">flags</th> </tr> </thead> <tbody> <tr> <td style="background-color: #d9e1f2;"><b>duration</b></td> <td>duration{+ = =}&lt;RELTIME&gt;</td> </tr> <tr> <td style="background-color: #d9e1f2;"><b>endtime</b></td> <td>endtime{+ = =}&lt;RELTIME&gt; or endtime=&lt;ABSTIME&gt;</td> </tr> <tr> <td style="background-color: #d9e1f2;"><b>hostexp</b></td> <td>hostexp{+ = =}&lt;node&gt;[,&lt;node&gt;]</td> </tr> <tr> <td style="background-color: #d9e1f2;"><b>variable</b></td> <td>variable[+=key1=val1 -key_to_remove]</td> </tr> <tr> <td style="background-color: #d9e1f2;"><b>reqtaskcount</b></td> <td>reqtaskcount{+ = =}&lt;TASKCOUNT&gt;</td> </tr> <tr> <td style="background-color: #d9e1f2;"><b>starttime</b></td> <td>starttime{+ = =}&lt;RELTIME&gt; or starttime=&lt;ABSTIME&gt;</td> </tr> </tbody> </table>	flags		<b>duration</b>	duration{+ = =}<RELTIME>	<b>endtime</b>	endtime{+ = =}<RELTIME> or endtime=<ABSTIME>	<b>hostexp</b>	hostexp{+ = =}<node>[,<node>]	<b>variable</b>	variable[+=key1=val1 -key_to_remove]	<b>reqtaskcount</b>	reqtaskcount{+ = =}<TASKCOUNT>	<b>starttime</b>	starttime{+ = =}<RELTIME> or starttime=<ABSTIME>
flags															
<b>duration</b>	duration{+ = =}<RELTIME>														
<b>endtime</b>	endtime{+ = =}<RELTIME> or endtime=<ABSTIME>														
<b>hostexp</b>	hostexp{+ = =}<node>[,<node>]														
<b>variable</b>	variable[+=key1=val1 -key_to_remove]														
<b>reqtaskcount</b>	reqtaskcount{+ = =}<TASKCOUNT>														
<b>starttime</b>	starttime{+ = =}<RELTIME> or starttime=<ABSTIME>														
<b>Description</b>	Modify aspects of a reservation.														

**-m****Example**

```
> mrsvctl -m duration=2:00:00 system.1
```

*Moab sets the duration of reservation `system.1` to be exactly two hours, thus modifying the endtime of the reservation.*

```
> mrsvctl -m starttime+=5:00:00 system.1
```

*Moab advances the starttime of `system.1` five hours from its current starttime (without modifying the duration of the reservation).*

```
> mrsvctl -m endtime-=5:00:00 system.1
```

*Moab moves the endtime of reservation `system.1` ahead five hours from its current endtime (without modifying the starttime; thus, this action is equivalent to modifying the duration of the reservation).*

```
> mrsvctl -m starttime=15:00:00_7/6/08 system.1
```

*Moab sets the starttime of reservation `system.1` to 3:00 p.m. on July 6, 2008.*

```
> mrsvctl -m starttime-=5:00:00 system.1
```

*Moab moves the starttime of reservation `system.1` ahead five hours.*

```
> mrsvctl -m starttime+=5:00:00 system.1
```

*Moab moves the starttime of reservation `system.1` five hours from the current time.*

```
> mrsvctl -m -duration+=5:00:00 system.1
```

*Moab extends the duration of `system.1` by five hours.*

```
> mrsvctl -m flags+=ADVRES system.1
```

*Moab adds the flag `ADVRES` to reservation `system.1`.*

**-m**

```
> mrsvctl -m variable+key1=val1 system.1
```

*Moab adds the variable `key1` with the value `key2` to `system.1`.*

```
> mrsvctl -m variable+=key1=val1 variable+=key2=val2 system.1
```

*Moab adds the variable `key1` with the value `val1`, and variable `key2` with `val2` to `system.1`. (Note that each variable flag requires a distinct `-m` entry.)*

```
> mrsvctl -m variable-=key1 system.1
```

*Moab deletes the variable `key1` from `system.1`.*

```
> mrsvctl -m variable-=key1 -m variable-=key2 system.1
```

*Moab deletes the variables `key1` and `key2` from `system.1`.*

**-m****Notes:**

- Modifying the starttime does not change the duration of the reservation, so the endtime changes as well. The starttime can be changed to be before the current time, but if the change causes the endtime to be before the current time, the change is not allowed.
- Modifying the endtime changes the duration of the reservation as well (and vice versa). An endtime *cannot* be placed before the starttime or before the current time.
- Duration cannot be negative.
- The += and -= operators operate on the time of the reservation (starttime+=5 adds five seconds to the current reservation starttime), while + and - operate on the current time (starttime+5 sets the starttime to five seconds from now).
- If the starttime or endtime specified is before the current time without a date specified, it is set to the next time that fits the command. To force the date, add the date as well. For the following examples, assume that the current time is 9:00 a.m. on March 1, 2007.

```
> mrsvctl -m starttime=8:00:00_3/1/07 system.1
```

*Moab moves system.1's starttime to 8:00 a.m., March 1.*

```
> mrsvctl -m starttime=8:00:00 system.1
```

*Moab moves system.1's starttime to 8:00 a.m., March 2.*

```
> mrsvctl -m endtime=7:00:00 system.1
```

*Moab moves system.1's endtime to 7:00 a.m., March 3. This happens because the endtime must also be after the starttime, so Moab continues searching until it has found a valid time that is in the future and after the starttime.*

```
> mrsvctl -m endtime=7:00:00_3/2/07 system.1
```

*Moab will return an error because the endtime cannot be before the starttime.*

**-n****Name**

NAME

**Format**

&lt;STRING&gt;

-n	
<b>Description</b>	<p>Name for new reservation.</p> <div style="border: 1px solid #004a87; border-radius: 5px; padding: 5px; margin-bottom: 5px;"> <p><b>i</b> If no name is specified, the reservation name is set to first name listed in ACL or SYSTEM if no ACL is specified.</p> </div> <div style="border: 1px solid #004a87; border-radius: 5px; padding: 5px;"> <p><b>i</b> Reservation names may not contain whitespace.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px;"> <pre>mrsvctl -c -h node01 -n John</pre> <p><i>Moab will create a reservation on node01 with the name John.</i></p> </div>

-o	
<b>Name</b>	OWNER
<b>Format</b>	<CREDTYPE>:<CREDID>
<b>Description</b>	Specifies the owner of a reservation. See <a href="#">Reservation Ownership</a> for more information.
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px;"> <pre>mrsvctl -c -h node01 -o USER:user1</pre> <p><i>Moab creates a reservation on node01 owned by user1.</i></p> </div>

-p	
<b>Name</b>	PARTITION
<b>Format</b>	<STRING>
<b>Description</b>	Only allocate resources from the specified partition
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px;"> <pre>mrsvctl -c -p switchB -t 14</pre> <p><i>Moab will allocate 14 tasks from the switchB partition.</i></p> </div>

-P	
<b>Name</b>	PROFILE
<b>Format</b>	<STRING>
<b>Description</b>	Indicates the <a href="#">reservation profile</a> to load when creating this reservation
<b>Example</b>	<pre>mrsvctl -c -P testing2 -t 14</pre> <p><i>Moab will allocate 14 tasks to a reservation defined by the testing2 reservation profile.</i></p>

-q	
<b>Name</b>	QUERY
<b>Format</b>	<RSV_ID> — The <code>-r</code> option accepts <a href="#">x: node regular expressions</a> and <a href="#">r: node range expressions</a> (asterisks (*) are supported wildcards as well).
<b>Description</b>	Get diagnostic information or list all completed reservations. The command gathers information from the Moab cache which prevents it from interrupting the scheduler, but the <code>--blocking</code> option can be used to bypass the cache and interrupt the scheduler.
<b>Example</b>	<pre>mrsvctl -q ALL</pre> <p><i>Moab will query reservations.</i></p> <pre>mrsvctl -q system.1</pre> <p><i>Moab will query the reservation system.1.</i></p>

-r	
<b>Name</b>	RELEASE
<b>Format</b>	<RSV_ID> — The <code>-r</code> option accepts <a href="#">x: node regular expressions</a> and <a href="#">r: node range expressions</a> (asterisks (*) are supported wildcards as well).

-r	
<b>Description</b>	Releases the specified reservation.
<b>Example</b>	<pre>&gt; mrsvctl -r system.1</pre> <p><i>Moab will release reservation system.1.</i></p> <pre>&gt; mrsvctl -r -g idle</pre> <p><i>Moab will release all idle job reservations.</i></p>

-R	
<b>Name</b>	RESOURCES
<b>Format</b>	<p>&lt;tid&gt; or            &lt;RES&gt;=&lt;VAL&gt;[{ + ;}&lt;RES&gt;=&lt;VAL&gt;]...</p> <p>Where &lt;RES&gt; is one of the following:            PROCS,            MEM,            DISK,            SWAP,            GRES</p>
<b>Default</b>	PROCS=-1
<b>Description</b>	<p>Specifies the resources to be reserved per task. (-1 indicates all resources on node)</p> <div style="border: 1px solid #4a7c9c; padding: 5px; background-color: #e6f2ff;"> <p><b>i</b> For GRES resources, &lt;VAL&gt; is specified in the format &lt;GRESNAME&gt;[:&lt;COUNT&gt;]</p> </div>
<b>Example</b>	<pre>&gt; mrsvctl -c -R MEM=100;PROCS=2 -t 2</pre> <p><i>Moab will create a reservation for two tasks with the specified resources.</i></p>

-S	
<b>Name</b>	STARTTIME
<b>Format</b>	[HH[:MM[:SS]]] [_MO[/DD[/YY]]] or +[[[DD:]HH:]MM:]SS
<b>Default</b>	[NOW]
<b>Description</b>	Absolute or relative time reservation will start. STARTTIME also supports an epoch timestamp.
<b>Example</b>	<pre>&gt; mrsvctl -c -t ALL -s 3:00:00_4/4/04</pre> <p><i>Moab will create a reservation on all system resources at 3:00 am on April 4, 2004</i></p> <pre>&gt; mrsvctl -c -h node01 -s +5:00</pre> <p><i>Moab will create a reservation in 5 minutes on node01</i></p> <pre>&gt; mrsvctl -m -s -=5:00 system.1</pre> <p><i>This will decrement the start time by 5 minutes.</i></p>

-S	
<b>Name</b>	SET ATTRIBUTE
<b>Format</b>	<ATTR>=<VALUE> where <ATTR> is one of aaccount — Accountable account agroup — accountable group aqos — accountable QoS auser — accountable user reqarch — required architecture reqmemory — required node memory - in MB reqos — required operating system rsvaccesslist — comma-delimited list of reservations or reservation groups which can be accessed by this reservation request. Because each reservation can access all other reservations by default, you should make any reservation with a specified rsvaccesslist exclusive by setting the <a href="#">-E on page 245</a> flag. This setting gives the otherwise exclusive reservation access to reservations specified in the list.

-S	
<b>Description</b>	Specifies a reservation attribute will be used to create this reservation
<b>Example</b>	<pre>&gt; mrsvctl -c -h node01 -S aqos=high</pre> <p><i>Moab will create a reservation on node01 and will use the QOS high as the accountable credential</i></p>

-t	
<b>Name</b>	TASKS
<b>Format</b>	<INTEGER>[-<INTEGER>]
<b>Description</b>	<p>Specifies the number of tasks to reserve. ALL indicates all resources available should be reserved.</p> <div style="border: 1px solid #005596; padding: 5px; background-color: #e6f2ff;"> <p><b>i</b> If the task value is set to ALL, Moab applies the reservation regardless of existing reservations and exclusive issues. If an integer is used, Moab only allocates accessible resources. If a range is specified Moab attempts to reserve the maximum number of tasks, or at least the minimum.</p> </div>
<b>Example</b>	<pre>&gt; mrsvctl -c -t ALL</pre> <p><i>Moab will create a reservation on all resources.</i></p> <pre>&gt; mrsvctl -c -t 3</pre> <p><i>Moab will create a reservation for three tasks.</i></p> <pre>&gt; mrsvctl -c -t 3-10 -E</pre> <p><i>Moab will attempt to reserve 10 tasks but will fail if it cannot get at least three.</i></p>

-T	
<b>Name</b>	TRIGGER

-T	
<b>Format</b>	<STRING>
<b>Description</b>	Comma-delimited reservation trigger list following format described in the trigger format section of the reservation configuration overview. See <a href="#">Creating a trigger on page 748</a> for more information.
	 To cancel a standing reservation with a trigger, the <b>SRCFG</b> parameter's attribute <a href="#">DEPTH</a> must be set to <i>0</i> .
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">&gt; mrsvctl -c -h node01 -T offset=200,etype=start,atype=exec,action=/tmp/email.sh</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"><i>Moab will create a reservation on node01 and fire the script /tmp/email.sh 200 seconds after it starts</i></p>

-V	
<b>Name</b>	VARIABLE
<b>Format</b>	<name>[=<value>][[:<name>[=<value>]]...]
<b>Description</b>	Semicolon-delimited list of variables that will be set when the reservation is created (See <a href="#">About trigger variables on page 775</a> for more information.). Names with no values will simply be set to <i>TRUE</i> .
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">&gt; mrsvctl -c -h node01 -V \$T1=mac;var2=18.19</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"><i>Moab will create a reservation on node01 and set \$T1 to mac and var2 to 18.19.</i></p>
	 For information on modifying a variable on a reservation, see <a href="#">MODIFY</a> .

-X	
<b>Name</b>	JOBLIST
<b>Format</b>	-x <jobs to be excluded>

-x	
<b>Description</b>	The <code>-x</code> flag, when used with <code>-F ignjobrsv</code> , lets users create reservations but exclude certain nodes that are running the listed jobs. The <code>-F</code> flag instructs <code>mrsvctl</code> to still consider nodes with current running jobs. The nodes are not listed directly.
<b>Example</b>	<pre>&gt; mrsvctl -c -t 5 -F ignjobrsv -x moab.5,moab.6</pre> <p><i>Moab will create the reservation while assigning the nodes. Nodes running jobs <code>moab5</code> and <code>moab6</code> will not be assigned to the reservation.</i></p>

## Parameters

RESERVATION ID	
<b>Format</b>	<STRING>
<b>Description</b>	The name of a reservation or a regular expression for several reservations.
<b>Example</b>	<pre>system*</pre> <p><i>Specifies all reservations starting with <code>system</code>.</i></p>

## Resource Allocation Details

When allocating resources, the following rules apply:

- When specifying tasks, each task defaults to one full compute node unless otherwise specified using the `-R` specification
- When specifying tasks, the reservation will not be created unless all requested resources can be allocated. (This behavior can be changed by specifying `-F besteffort`)
- When specifying tasks or hosts, only nodes in an idle or running state will be considered. (This behavior can be changed by specifying `-F ignstate`)

## Reservation Timeframe Modification

Moab supports dynamically modifying the timeframe of existing reservations. This can be accomplished using the `mrsvctl -m` flag. By default, Moab will perform advanced boundary and resource access to verify that the modification does not result in an invalid scheduler state. However, in certain circumstances administrators may wish to FORCE the modification in spite of any access violations. This can be done using the switch `mrsvctl -m --flags=force` which forces Moab to bypass any access verification and force the change through.

## Extending a reservation by modifying the endtime

The following increases the endtime of a reservation using the += tag:

```
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:35:57  1:11:35:57  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m endtime+=24:00:00 system.1
endtime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:35:22  2:11:35:22  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
```

The following increases the endtime of a reservation by setting the endtime to an absolute time:

```
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:33:18  1:11:33:18  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m endtime=0_11/20 system.1
endtime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:33:05  2:11:33:05  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
```

## Extending a reservation by modifying the duration

The following increases the duration of a reservation using the += tag:

```
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:28:46  1:11:28:46  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m duration+=24:00:00 system.1
duration for rsv 'system.1' changed
>$ showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:28:42  2:11:28:42  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
```

The following increases the duration of a reservation by setting the duration to an absolute time:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:26:41   1:11:26:41  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m duration=48:00:00 system.1
duration for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:26:33   2:11:26:33  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located

```

## Shortening a reservation by modifying the endtime

The following modifies the endtime of a reservation using the `--` tag:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:15:51   2:11:15:51  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m endtime-=24:00:00 system.1
endtime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:15:48   1:11:15:48  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located

```

The following modifies the endtime of a reservation by setting the endtime to an absolute time:

```

$ showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:14:00   2:11:14:00  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m endtime=0_11/19 system.1
endtime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:13:48   1:11:13:48  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located

```

## Shortening a reservation by modifying the duration

The following modifies the duration of a reservation using the `--` tag:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:12:20   2:11:12:20  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m duration-=24:00:00 system.1
duration for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:12:07   1:11:12:07  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located

```

The following modifies the duration of a reservation by setting the duration to an absolute time:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:10:57   2:11:10:57  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m duration=24:00:00 system.1
duration for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:10:50   1:11:10:50  1:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located

```

## Modifying the starttime of a reservation

The following increases the starttime of a reservation using the += tag:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:08:30   2:11:08:30  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m starttime+=24:00:00 system.1
starttime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     1:11:08:22  3:11:08:22  2:00:00:00  1/2      Sun Nov 19
00:00:00
1 reservation located

```

The following decreases the starttime of a reservation using the -= tag:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     11:07:04   2:11:07:04  2:00:00:00  1/2      Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m starttime-=24:00:00 system.1
starttime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
system.1          User -     -12:53:04  1:11:06:56  2:00:00:00  1/2      Fri Nov 17
00:00:00
1 reservation located

```

The following modifies the starttime of a reservation using an absolute time:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P  StartTime
system.1          User -    11:05:31  2:11:05:31  2:00:00:00  1/2  Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m starttime=0_11/19 system.1
starttime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P  StartTime
system.1          User -    1:11:05:18  3:11:05:18  2:00:00:00  1/2  Sun Nov 19
00:00:00
1 reservation located

```

The following modifies the starttime of a reservation using an absolute time:

```

$> showres
ReservationID      Type S      Start      End      Duration  N/P  StartTime
system.1          User -    11:04:04  2:11:04:04  2:00:00:00  1/2  Sat Nov 18
00:00:00
1 reservation located
$> mrsvctl -m starttime=0_11/17 system.1
starttime for rsv 'system.1' changed
$> showres
ReservationID      Type S      Start      End      Duration  N/P  StartTime
system.1          User -    -12:56:02  1:11:03:58  2:00:00:00  1/2  Fri Nov 17
00:00:00
1 reservation located

```

## Examples

- [Basic Reservation on page 262](#)
- [System Maintenance Reservation on page 262](#)
- [Explicit Task Description on page 263](#)
- [Dynamic Reservation Modification on page 263](#)
- [Reservation Modification on page 263](#)
- [Allocating Reserved Resources on page 263](#)
- [Modifying an Existing Reservation on page 263](#)

### Example 3-36: Basic Reservation

Reserve two nodes for use by users john and mary for a period of 8 hours starting in 24 hours

```

> mrsvctl -c -a USER=john,USER=mary -starttime +24:00:00 -duration 8:00:00 -t 2
reservation 'system.1' created

```

### Example 3-37: System Maintenance Reservation

Schedule a system wide reservation to allow a system maintenance on Jun 20, 8:00 AM until Jun 22, 5:00 PM.

```
% mrsvctl -c -s 8:00:00_06/20 -e 17:00:00_06/22 -h ALL
reservation 'system.1' created
```

**Example 3-38: Explicit Task Description**

Reserve one processor and 512 MB of memory on nodes `node003` through node `node006` for members of the group `staff` and jobs in the interactive class

```
> mrsvctl -c -R PROCS=1,MEM=512 -a GROUP=staff,CLASS=interactive -h 'node00[3-6]'
reservation 'system.1' created
```

**Example 3-39: Dynamic Reservation Modification**

Modify reservation `john.1` to start in 2 hours, run for 2 hours, and include `node02` in the hostlist.

```
> mrsvctl -m starttime=+2:00:00,duration=2:00:00,HostExp+=node02
Note: hosts added to rsv system.3
```

**Example 3-40: Reservation Modification**

Remove user `John`'s access to reservation `system.1`

```
> mrsvctl -m -a USER=John system.1 --flags=unset
successfully changed ACL for rsv system.1
```

**Example 3-41: Allocating Reserved Resources**

Allocate resources for group `dev` which are [exclusive](#) except for resources found within reservations `myrinet.3` or `john.6`

```
> mrsvctl -c -E -a group=dev,rsv=myrinet.3,rsv=john.6 -h 'node00[3-6]'
reservation 'dev.14' created
```

Create exclusive network reservation on racks 3 and 4

```
> mrsvctl -c -E -a group=ops -g network -f rack3 -h ALL
reservation 'ops.1' created
> mrsvctl -c -E -a group=ops -g network -f rack4 -h ALL
reservation 'ops.2' created
```

Allocate 64 nodes for 2 hours to new reservation and grant access to reservation `system.3` and all reservations in the reservation group `network`

```
> mrsvctl -c -E -d 2:00:00 -a group=dev -t 64 -S rsvaccesslist=system.3,network
reservation 'system.23' created
```

Allocate 4 nodes for 1 hour to new reservation and grant access to idle job reservations

```
> mrsvctl -c -E -d 1:00:00 -t 4 -S rsvaccesslist=idle
reservation 'system.24' created
```

**Example 3-42: Modifying an Existing Reservation**

Remove user `john` from reservation ACL

```
> mrsvctl -m -a USER=john system.1 --flags=unset
successfully changed ACL for rsv system.1
```

### Change reservation group

```
> mrsvctl -m RSVGROUP=network ops.4
successfully changed RSVGROUP for rsv ops.4
```

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [Admin Reservation Overview](#)
- [showres](#)
- [mdiag -r](#)
- [mshow -a](#) command to identify available resources
- [job to rsv binding](#)

## mschedctl

### Synopsis

[mschedctl -A](#) '<MESSAGE>'

[mschedctl -c](#) message *messagestring* [-o *type:val*]

[mschedctl -c](#) trigger *triggerid* -o *type:val*

[mschedctl -d](#) trigger:*triggerid*

[mschedctl -d](#) message:*index*

[mschedctl -f](#) {all|fairshare|usage}

[mschedctl -k](#)

[mschedctl -l](#) {config|gmetric|gres|message|opsys|trigger|trans} [--flags=verbose] [--xml]

[mschedctl -L](#) [*LOGLEVEL*]

[mschedctl -m](#) config *string* [-e] [--flags=persistent]

[mschedctl -m](#) trigger *triggerid* *attr=val* [,*attr=val...*]

[mschedctl -q](#) mschedctl -q pactions --xml

[mschedctl -p](#)

[mschedctl -r](#) [*resumetime*]

[mschedctl -R](#)

[mschedctl -s](#) [*STOPITERATION*]

[mschedctl -S](#) [*STEPITERATION*]

## Overview

The `mschedctl` command controls various aspects of scheduling behavior. It is used to manage scheduling activity, shutdown the scheduler, and create resource trace files. It can also evaluate, modify, and create parameters, triggers, and messages.

**i** With many flags, the `--msg=<MSG>` option can be specified to annotate the action in the [event log](#).

## Format

-A - ANNOTATE	
<b>Format</b>	<STRING>
<b>Description</b>	Report the specified parameter modification to the event log and annotate it with the specified message. The <b>RECORDEVENTLIST</b> parameter must be set in order for this to work.
<b>Example</b>	<pre>mschedctl -A 'increase logging' -m 'LOGLEVEL 6'</pre> <p><i>Adjust the <b>LOGLEVEL</b> parameter and record an associated message.</i></p>
-c - CREATE	
<b>Format</b>	One of: <ul style="list-style-type: none"> <li>• <b>message</b> &lt;STRING&gt; [-o &lt;TYPE&gt;:&lt;VAL&gt;]</li> <li>• <b>trigger</b> &lt;TRIGSPEC&gt; -o &lt;OBJECTTYPE&gt;:&lt;OBJECTID&gt;</li> <li>• <b>gevent</b> -n &lt;NAME&gt; [-m &lt;message&gt;]</li> </ul> <p>where &lt;ATTR&gt; is one of account, duration, ID, messages, profile, reqresources, resources, rsvprofile, starttime, user, or variables</p>
<b>Description</b>	Create a message, trigger, or gevent and attach it to the specified object. To create a trigger on a default object, use the Moab configuration file ( <code>moab.cfg</code> ) rather than the <code>mschedctl</code> command.

**-c - CREATE**

**Example**

```
mschedctl -c message tell the admin to be nice
```

*Create a message on the system table.*

```
mschedctl -c trigger EType=start,AType=exec,Action="/tmp/email $OWNER $TIME" -o rsv:system.1
```

*Create a trigger linked to system.1.*



Creating triggers on default objects via `mschedctl -c trigger` does not propagate the triggers to individual objects. To propagate triggers to all objects, the triggers must be created within the `moab.cfg` file; for example: `NODECFG[DEFAULT] TRIGGER`.

```
mschedctl -c gevent -n diskfailure -m "node=n4"
```

*Create a gevent indicating a disk failure on the node labeled n4.*

**-d - DESTROY**

**Format**

One of:

- **trigger:**<TRIGID>
- **message:**<INDEX>

**Description**

Delete a trigger or message.

**Example**

```
mschedctl -d trigger:3
```

*Delete trigger 3.*

```
mschedctl -d message:5
```

*Delete message with index 5.*

-f - FLUSH	
<b>Format</b>	{all fairshare usage}
<b>Description</b>	Reset all internally-stored Moab Scheduler statistics to the initial start-up state as of the time the command was executed.
<b>Example</b>	<pre>mschedctl -f usage</pre> <p><i>Flush usage statistics.</i></p>

-k - KILL	
<b>Description</b>	Stop scheduling and exit the scheduler
<b>Example</b>	<pre>mschedctl -k</pre> <p><i>Kill the scheduler.</i></p>

-l - LIST	
<b>Format</b>	{config   gmetric   gres   message opsys   trans   trigger} [--flags=verbose] [--xml] <div style="border: 1px solid #000; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p><b>i</b> Using the <code>--xml</code> argument with the <code>trans</code> option returns XML that states if the queried TID is valid or not.</p> </div>
<b>Default</b>	config
<b>Description</b>	List the generic metrics, generic resources, scheduler configuration, system messages, operating systems, triggers, or transactions.

### -l - LIST

**Example**

```
mschedctl -l config
```

*List system parameters.*

```
mschedctl -l gmetric
```

*List all configured generic metrics.*

```
mschedctl -l gres
```

*List all configured generic resources.*

```
mschedctl -l message
```

*List all system messages.*

```
mschedctl -l opsys
```

*List all recognized operating systems*

```
mschedctl -l trans 1
```

*List transaction id 1.*

```
mschedctl -l trigger
```

*List triggers.*

### -L - LOG

**Format**

<INTEGER>

**Default**

7

**Description**

Create a temporary log file with the specified loglevel.

**Example**

```
mschedctl -L 7
```

*Create temporary log file with naming convention  
<logfile>.YYYYMMDDHHMMSS.*

-m - MODIFY	
<b>Format</b>	<p>One of:</p> <ul style="list-style-type: none"> <li>• <b>config</b> [&lt;STRING&gt;] <ul style="list-style-type: none"> <li>[-e]</li> <li>[--flags=pers]</li> </ul>           &lt;STRING&gt; is any string which would be acceptable in moab.cfg <ul style="list-style-type: none"> <li>◦ If no string is specified, &lt;STRING&gt; is read from STDIN.</li> <li>◦ If -e is specified, the configuration string will be evaluated for correctness but no configuration changes will take place. Any issues with the provided string will be reported to STDERR.</li> <li>◦ If --flags=persistent is specified, the Moab configuration files (moab.cfg and moab.dat) are modified.</li> </ul> </li> <li>• <b>trigger</b>:&lt;TRIGID&gt; &lt;ATTR&gt;=&lt;VAL&gt;</li> </ul> <p>where &lt;ATTR&gt; is one of action, atype, etype, iscomplete, oid, otype, offset, or threshold</p>
<b>Description</b>	Modify a system parameter or trigger.
<b>Example</b>	<pre>mschedctl -m config LOGLEVEL 9</pre> <p><i>Change the system loglevel to 9.</i></p> <pre>mschedctl -m trigger:2 AType=exec,Offset=200,OID=system.1</pre> <p><i>Change aspects of trigger 2.</i></p>
-p - PAUSE	
<b>Description</b>	Disable scheduling but allow the scheduler to update its cluster and workload state information.
<b>Example</b>	<pre>mschedctl -p</pre>
-q QUERY PENDING ACTIONS	
<b>Default</b>	mschedctl -q pactions --xml
<b>Description</b>	A way to view pending actions. Only an XML request is valid. Pending actions can be VMs or system jobs.

### -q QUERY PENDING ACTIONS

<b>Example</b>	<code>mschedctl -q pactions --xml</code>
----------------	------------------------------------------

### -R - RECYCLE

<b>Description</b>	Recycle scheduler immediately (shut it down and restart it using the original execution environment and command line arguments).
<b>Example</b>	<pre>mschedctl -R</pre> <p><i>Recycle scheduler immediately.</i></p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p> To restart Moab with its last known scheduler state, use: <code>mschedctl -R savestate</code></p> </div>

### -r - RESUME

<b>Format</b>	<code>mschedctl -r [[HH:[MM:]]SS]</code>
<b>Default</b>	0
<b>Description</b>	Resume scheduling in the specified amount of time (or immediately if none is specified).
<b>Example</b>	<pre>mschedctl -r</pre> <p><i>Resume scheduling immediately.</i></p>

### -s - STOP

<b>Format</b>	<code>&lt;INTEGER&gt;</code>
<b>Default</b>	0
<b>Description</b>	Suspend/stop scheduling at specified iteration (or at the end of the current iteration if none is specified). If the letter I follows <code>&lt;ITERATION&gt;</code> , Moab will not process client requests until this iteration is reached.

-s - STOP	
<b>Example</b>	<pre>mschedctl -s 100I</pre> <p><i>Stop scheduling at iteration 100 and ignore all client requests until then.</i></p>
-S - STEP	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Step the specified number of iterations (or to the next iteration if none is specified) and suspend scheduling. If the letter I follows <ITERATION>, Moab will not process client requests until this iteration is reached.
<b>Example</b>	<pre>mschedctl -S</pre> <p><i>Step to the next iteration and stop scheduling.</i></p>

## Examples

*Example 3-43: Shutting down the Scheduler*

```
mschedctl -k
scheduler will be shutdown immediately
```

### Related topics

- Moab Client Installation - explains how to distribute this command to client nodes

## mshow

### Synopsis

```
mshow [-a] [-q jobqueue=active]
```

### Overview

The `mshow` command displays various diagnostic messages about the system and job queues.

## Arguments

Flag	Description
<a href="#">-a</a>	<a href="#">AVAILABLE RESOURCES</a>
<b>-q</b> [<QUEUENAME>]	Displays the job queues.

## Format

AVAILABLE RESOURCES	
<b>Format</b>	Can be combined with <code>--flags=[tid verbose future]</code> <code>--format=xml</code> and/or <code>-w</code>
<b>Description</b>	Display available resources.
<b>Example</b>	<pre>&gt; mshow -a -w user=john --flags=tid --format=xml</pre> <p><i>Show resources available to john in XML format with a transaction id. See <a href="#">mshow -a</a> for details.</i></p>

JOB QUEUE	
<b>Format</b>	<QUEUENAME>, where the queue name is one of: active, eligible, or blocked. Job queue names can be delimited by a comma to display multiple queues. If no job queue name is specified, mshow displays all job queues.
<b>Description</b>	Displays the job queues. If a job queue name is specified, mshow shows only that job queue.
<b>Example</b>	<pre>&gt; mshow -q active,blocked [Displays all jobs in the active and blocked queues] ...</pre>

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mshow -a](#) command to show available resources

### mshow -a

## Synopsis

`mshow -a [-i] [-o] [-T] [-w where] [-x] [--xml]`

## Overview

The `mshow -a` command allows for querying of available system resources.

## Arguments

<code>[-i]</code>	INTERSECTION
<code>[-o]</code>	NO AGGREGATE
<code>[-T]</code>	TIMELOCK
<code>[-w]</code>	WHERE
<code>[-x]</code>	EXCLUSIVE

Table 3-4: Argument Format

--flags	
<b>Name</b>	Flags
<b>Format</b>	<code>--flags=[ future   policy   tid   summary   verbose ]</code>
<b>Description</b>	<p><b>future</b> will return resources available immediately and available in the future.</p> <p><b>policy (Deprecated. May be removed in a future release.)</b> will apply charging policies to determine the total cost of each reported solution (only enabled for XML responses).</p> <p><b>summary</b> will assign all jointly allocated transactions as dependencies of the first transaction reported.</p> <p><b>tid</b> will associate a transaction id with the reported results.</p> <p><b>verbose</b> will return diagnostic information.</p>
<b>Example</b>	<pre>&gt; mshow -a -w user=john --flags=tid --xml</pre> <p><i>Show resources available to john in XML format with a transaction ID.</i></p>

--xml	
<b>Name</b>	XML
<b>Format</b>	--xml
<b>Description</b>	Report results in XML format.
<b>Example</b>	<pre>&gt; mshow -a -w user=john --flags=tid --xml</pre> <p><i>Show resources available to john in XML format with a transaction ID.</i></p>

-i	
<b>Name</b>	INTERSECTION
<b>Description</b>	Specifies that an intersection should be performed during an <code>mshow -a</code> command with multiple requirements.

-o	
<b>Name</b>	NO AGGREGATE
<b>Description</b>	Specifies that the results of the command <code>mshow -a</code> with multiple requirements should not be aggregated together.

-T	
<b>Name</b>	TIMELock
<b>Description</b>	Specifies that the multiple requirements of an <code>mshow -a</code> command should be timelocked.
<b>Example</b>	<pre>&gt; mshow -a -w minprocs=1,os=linux,duration=1:00:00 \ -w minprocs=1,os=aix,duration=10:00 \ --flags=tid,future -x -T</pre>

-w	
<b>Name</b>	WHERE
<b>Format</b>	Comma delimited list of <ATTR>=<VAL> pairs: <ATTR>=<VAL> [,<ATTR>=<VAL>]... <div style="border: 1px solid #004a7c; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p><b>i</b> If any of the &lt;ATTR&gt;=&lt;VAL&gt; pairs contains a sub-list that is also comma delimited, the entire -w string must be wrapped in single quotations with the sub-list expression wrapped in double quotations. See the example below.</p> </div> Attributes are listed below in <a href="#">table 2</a> .
<b>Description</b>	Add a Where clause to the current command (currently supports up to six co-allocation clauses).
<b>Example</b>	<pre>&gt; mshow -a -w minprocs=2,duration=1:00:00 -w nodemem=512,duration=1:00:00</pre> <p><i>Moab returns a list of all nodes with at least 2 processors and one hour duration or with a memory of 512 and a duration of one hour.</i></p> <pre>&gt; mshow -a -w nodefeature=!vmware:gpfs --flags=future</pre> <p><i>Moab returns a list of all nodes that do not contain the vmware feature but that do contain the gpfs feature.</i></p> <pre>&gt; mshow -a -w 'duration=INFINITY,"excludehostlist=n01,n12,n23"'</pre> <p><i>Moab returns a list of all nodes with a duration of INFINITY, except for nodes named n01, n12, and n23.</i>  <i>Note the use of single quotations containing the entire -w string and the use of double quotations containing the excludehostlist attribute.</i></p>

-x	
<b>Name</b>	EXCLUSIVE
<b>Description</b>	Specifies that the multiple requirements of an mshow -a command should be exclusive (i.e. each node may only be allocated to a single requirement)
<b>Example</b>	<pre>&gt; mshow -a -w minprocs=1,os=linux -w minprocs=1,os=aix --flags=tid -x</pre>

Table 3-5: Request Attributes

Name	Description
<b>account</b>	The account credential of the requestor
<b>acl</b>	ACL to attach to the reservation  This ACL must be enclosed in quotation marks. For example: <code>\$ mshow -a ... -w acl=\"user=john\" ...</code>
<b>arch</b>	Select only nodes with the specified architecture
<b>cal</b>	Select resources subject to the constraints of the specified global calendar
<b>class</b>	The class credential of the requestor
<b>coalloc</b>	The co-allocation group of the specific Where request (can be any string but must match co-allocation group of at least one other Where request)  <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF;"> <span style="font-size: 1.2em; font-weight: bold; color: #0070C0;">i</span> The number of tasks requested in each Where request must be equal whether this taskcount is specified via <code>minprocs</code>, <code>mintasks</code>, or <code>gres</code>.                     </div>
<b>count</b>	The number of profiles to apply to the resource request
<b>displaymode</b>	Possible value is <code>future</code> . (Example: <code>displaymode=future</code> ). Constrains how results are presented; setting <code>future</code> evaluates which resources are available now and which resources will be available in the future that match the requested attributes.
<b>duration</b>	The duration for which the resources will be required in format <code>[ [DD:]HH:]MM:]SS</code>
<b>excludehostlist</b>	Do not select any nodes from the given list. The list must be comma delimited.  <div style="border: 1px dashed #ccc; padding: 10px;"> <pre>&gt; mshow -a -w 'duration=INFINITY,"excludehostlist=n01,n12,n23"'</pre> <p><i>Moab returns a list of all nodes with a duration of <b>INFINITY</b>, except for nodes named <b>n01</b>, <b>n12</b>, and <b>n23</b>.</i></p> <p><i>Note the use of single quotations to contain the entire <code>-w</code> string, and the use of double quotations containing the <b>excludehostlist</b> attribute.</i></p> </div>
<b>gres</b>	Select only nodes which possess the specified generic resource
<b>group</b>	The group credential of the requestor

Name	Description
<b>hostlist</b>	<p>Select only the specified resources. The list must be comma delimited.</p> <pre>&gt; mshow -a -w 'duration=INFINITY,"hostlist=n01,n12,n23"'</pre> <p><i>Moab returns a list of nodes from the selected hostlist that have a duration of <b>INFINITY</b>.</i></p> <p><i>Note the use of single quotations to contain the entire <code>-w</code> string, and the use of double quotations containing the <b>hostlist</b> attribute.</i></p>
<b>job</b>	Use the resource, duration, and credential information for the job specified as a resource request template
<b>jobfeature</b>	Select only resources which would allow access to jobs with the specified job features
<b>jobflags</b>	Select only resources which would allow access to jobs with the specified job flags. The <code>job-flags</code> attribute accepts a colon delimited list of multiple flags.
<b>label</b>	Removed label 1/2013 for DOC-16 Associate the specified label with all results matching this request
<b>minnodes</b>	Return only results with at least the number of nodes specified. If used with TID's, return only solutions with exactly minnodes nodes available
<b>minprocs</b>	Return only results with at least the number of processors specified. If used with TID's, return only solutions with exactly minprocs processors available
<b>mintasks</b>	<b>FORMAT:</b> <code>&lt;TASKCOUNT&gt;[@&lt;RESTYPE&gt;:&lt;COUNT&gt;[+&lt;RESTYPE&gt;:&lt;COUNT&gt;]...]</code> where <code>&lt;RESTYPE&gt;</code> is one of <code>procs</code> , <code>mem</code> , <code>disk</code> , or <code>swap</code> . Return only results with at least the number of tasks specified. If used with TID's, return only solutions with exactly mintasks available
<b>nodedisk</b>	Select only nodes with at least <code>nodedisk</code> MB of local disk configured
<b>nodefeature</b>	Select only nodes with all specified features present and nodes without all \! specified features using format <code>[ \! ] &lt;feature&gt; [ : [ \! ] &lt;feature&gt; ] ...</code> . You must set the <a href="#">future</a> flag when specifying node features.
<b>nodemem</b>	Select only nodes with at least <code>nodemem</code> MB of memory configured
<b>offset</b>	Select only resources which can be co-allocated with the specified time offset where offset is specified in the format <code>[ [ DD : ] HH : ] MM : ] SS</code>

Name	Description
<b>os</b>	Select only nodes with have, or can be provisioned to have, the specified operating system
<b>partition</b>	The partition in which the resources must be located
<b>policylevel</b>	Enable policy enforcement at the specified policy constraint level
<b>qos</b>	The qos credential of the requestor
<b>rsvprofile</b>	Use the specified profile if committing a resulting transaction id directly to a reservation
<b>starttime</b>	<p>Constrain the timeframe for the returned results by specifying one or more ranges using the format <code>&lt;STIME&gt;[-&lt;ENDTIME&gt;];&lt;STIME&gt;[-&lt;ENDTIME&gt;]</code> where each time is specified in the format in absolute, relative, or epoch time format (<code>[HH[:MM[:SS]]][_MO[/DD[/YY]]]</code>) or <code>+ [[DD:]HH:]MM:]SS</code> or <code>&lt;EPOCHTIME&gt;</code>).</p> <div style="border: 1px solid #004a87; border-radius: 10px; padding: 5px; background-color: #e6f2ff;"> <p> The starttime specified is not the exact time at which the returned range must start, but is rather the earliest possible time the range may start.</p> </div>
<b>taskmem</b>	Require <code>taskmem</code> MB of memory per task located
<b>tpn</b>	Require exactly <code>tpn</code> tasks per node on all discovered resources
<b>user</b>	The user credential of the requestor
<b>var</b>	Use associated variables in generating per transaction charging quotes
<b>variables</b>	Takes a string of the format <code>variables='var[=attr]';'var[=attr]'</code> and passes the variables onto the reservation when used in conjunction with <code>--flags=tid</code> and <code>mrsvctl -c -R &lt;tid&gt;</code> .
<b>vmusage</b>	Possible value is <code>vmcreate</code> . Moab will find resources for the job assuming it is a <code>vmcreate</code> job, and if <code>os</code> is also specified, Moab will look for a hypervisor capable of running a VM with the requested OS.

## Usage Notes

The `mshow -a` command allows for querying of available system resources. When combined with the `--flags=tid` option these available resources can then be placed into a packaged reservation (using [mrsvctl -c -R](#)). This allows system administrators to grab and reserve available resources for whatever reason, without conflicting with jobs or reservations that may be holding certain resources.

There are a few restrictions on which *<ATTR>* from the *-w* command can be placed in the same req: *minprocs*, *minnodes*, and *gres* are all mutually exclusive, only one may be used per *-w* request.

The allocation of available nodes will follow the global [NODEALLOCATIONPOLICY](#).

When the *-o* flag is not used, multi-request results will be aggregated. This aggregation will negate the use of offsets and request-specific starttimes.

The config parameter [RESOURCEQUERYDEPTH](#) controls the maximum number of options that will be returned in response to a resource query.

## Examples

### Example 3-44: Basic Compute Node Query and Reservation

```
> mshow -a -w duration=10:00:00,minprocs=1,os=AIX53,jobfeature=shared --
flags=tid,future
```

Partition	Tasks	Nodes	Duration	StartOffset	StartDate		
ALL	1	1	10:00:00	00:00:00	13:28:09_04/27	TID=4	ReqID=0
ALL	1	1	10:00:00	10:00:00	17:14:48_04/28	TID=5	ReqID=0
ALL	1	1	10:00:00	20:00:00	21:01:27_04/29	TID=6	ReqID=0

```
> mrsvctl -c -R 4
Note: reservation system.2 created
```

### Example 3-45: Mixed Processor and License Query

Select one node with 4 processors and 1 matlab license where the matlab license is only available for the last hour of the reservation. Also, select 16 additional processors which are available during the same timeframe but which can be located anywhere in the cluster. Group the resulting transactions together using transaction dependencies so only the first transaction needs to be committed to reserve all associated resources.

```
> mshow -a -i -o -x -w mintasks=1@PROCS:4,duration=10:00:00,coalloc=a \
-w gres=matlab,offset=9:00:00,duration=1:00:00,coalloc=a \
-w minprocs=16,duration=10:00:00 --flags=tid,future,summary
```

Partition	Tasks	Nodes	Duration	StartOffset	StartDate		
ALL	1	1	10:00:00	00:00:00	13:28:09_04/27	TID=4	ReqID=0
ALL	1	1	10:00:00	10:00:00	17:14:48_04/28	TID=5	ReqID=0
ALL	1	1	10:00:00	20:00:00	21:01:27_04/29	TID=6	ReqID=0

```
> mrsvctl -c -R 4

Note: reservation system.2 created
Note: reservation system.3 created
Note: reservation system.4 created
```

### Example 3-46: Request for Generic Resources

Query for a generic resource on a specific host (no processors, only a generic resource).

```

> mshow -a -i -x -o -w gres=dvd,duration=10:00,hostlist=node03 --flags=tid,future
Partition      Tasks  Nodes  StartOffset  Duration  StartDate
-----
ALL            1      1      00:00:00     00:10:00  11:33:25_07/27  TID=16
ReqID=0
ALL            1      1      00:10:00     00:10:00  11:43:25_07/27  TID=17
ReqID=0
ALL            1      1      00:20:00     00:10:00  11:53:25_07/27  TID=18
ReqID=0
> mrvsvctl -c -R 16
Note: reservation system.6 created
> mdiag -r system.6
Diagnosing Reservations
RsvID          Type Par  StartTime  EndTime  Duration Node Task
Proc
-----
-
system.6       User loc  -00:01:02  00:08:35  00:09:37  1  1
0
Flags: ISCLOSED
ACL:   RSV==system.6=
CL:   RSV==system.6
Accounting Creds: User:test
Task Resources: dvd: 1
Attributes (HostExp='^node03$')
Rsv-Group: system.6

```

#### Example 3-47: Allocation of Shared Resources

This example walks through a relatively complicated example in which a set of resources can be reserved to be allocated for shared requests. In the example below, the first `mshow` query looks for resources within an existing shared reservation. In the example, this first query fails because there is now existing reservation. The second query looks for resources within an existing shared reservation. In the example, this first query fails because there is now existing reservation. The second `mshow` request asks for resources outside of a shared reservation and finds the desired resources. These resources are then reserved as a shared pool. The third `mshow` request again asks for resources inside of a shared reservation and this time finds the desired resources.

```

> mshow -a -w duration=10:00:00,minprocs=1,os=AIX53,jobflags=ADVRES,jobfeature=shared
--flags=tid
Partition      Tasks  Nodes  Duration  StartOffset  StartDate
-----
> mshow -a -w duration=100:00:00,minprocs=1,os=AIX53,jobfeature=shared --flags=tid
Partition      Tasks  Nodes  Duration  StartOffset  StartDate
-----
ALL            1      1      100:00:00  00:00:00     13:20:23_04/27  TID=1  ReqID=0
> mrvsvctl -c -R 1
Note: reservation system.1 created
> mshow -a -w duration=10:00:00,minprocs=1,os=AIX53,jobflags=ADVRES,jobfeature=shared
--flags=tid
Partition      Tasks  Nodes  Duration  StartOffset  StartDate
-----
ALL            1      1      10:00:00   00:00:00     13:20:36_04/27  TID=2  ReqID=0
> mrvsvctl -c -R 2
Note: reservation system.2 created

```

*Example 3-48: Full Resource Query in XML Format*

The following command will report information on all available resources which meet at least the minimum specified processor and walltime constraints and which are available to the specified user. The results will be reported in XML to allow for easy system processing.

```

> mshow -a -w class=grid,minprocs=8,duration=20:00 --format=xml --flags=future,verbose

<Data>
  <Object>cluster</Object>
  <job User="john" time="1162407604"></job>
  <par Name="template">
    <range duration="Duration" nodecount="Nodes" proccount="Procs"
starttime="StartTime"></range>
  </par>
  <par Name="ALL" feasibleNodeCount="131" feasibleTaskCount="163">
    <range duration="1200" hostlist="opt-001:1,opt-024:1,opt-025:1,opt-027:2,opt-
041:1,opt-042:1,x86-001:1,P690-001:1,P690-021:1,P690-022:1"
      index="0" nodecount="10" proccount="8" reqid="0"
starttime="1162407604"></range>
    <range duration="1200" hostlist="opt-001:1,opt-024:1,opt-025:1,opt-027:2,opt-
039:1,opt-041:1,opt-042:1,x86-001:1,P690-001:1,P690-021:1,P690-022:1"
      index="0" nodecount="11" proccount="8" reqid="0"
starttime="1162411204"></range>
    <range duration="1200" hostlist="opt-001:1,opt-024:1,opt-025:1,opt-027:2,opt-
039:1,opt-041:1,opt-042:1,x86-001:1,x86-002:1,x86-004:1,
      x86-006:1,x86-013:1,x86-014:1,x86-015:1,x86-016:1,x86-037:1,P690-001:1,P690-
021:1,P690-022:1"
      index="0" nodecount="19" proccount="8" reqid="0"
starttime="1162425519"></range>
  </par>
  <par Name="SharedMem">
    <range duration="1200" hostlist="P690-001:1,P690-002:1,P690-003:1,P690-004:1,P690-
005:1,P690-006:1,P690-007:1,P690-008:1,P690-009:1,
      P690-010:1,P690-011:1,P690-012:1,P690-013:1,P690-014:1,P690-015:1,P690-
016:1,P690-017:1,P690-018:1,P690-019:1,P690-020:1,P690-021:1,
      P690-022:1,P690-023:1,P690-024:1,P690-025:1,P690-026:1,P690-027:1,P690-
028:1,P690-029:1,P690-030:1,P690-031:1,P690-032:1"
      index="0" nodecount="32" proccount="8" reqid="0"
starttime="1163122507"></range>
  </par>
  <par Name="64Bit">
    <range duration="1200" hostlist="opt-001:1,opt-024:1,opt-025:1,opt-027:2,opt-
039:1,opt-041:1,opt-042:1"
      index="0" nodecount="7" proccount="8" reqid="0"
starttime="1162411204"></range>
    <range duration="1200" hostlist="opt-001:1,opt-024:1,opt-025:1,opt-027:2,opt-
039:1,opt-041:1,opt-042:1,opt-043:1,opt-044:1,opt-045:1,
      opt-046:1,opt-047:1,opt-048:1,opt-049:1,opt-050:1"
      index="0" nodecount="15" proccount="8" reqid="0"
starttime="1162428996"></range>
    <range duration="1200" hostlist="opt-001:1,opt-006:1,opt-007:2,opt-008:2,opt-
009:2,opt-010:2,opt-011:2,opt-012:2,opt-013:2,opt-014:2,
      opt-015:2,opt-016:2,opt-017:2,opt-018:2,opt-019:2,opt-020:2,opt-021:2,opt-
022:2,opt-023:2,opt-024:2,opt-025:1,opt-027:2,opt-039:1,
      opt-041:1,opt-042:1,opt-043:1,opt-044:1,opt-045:1,opt-046:1,opt-047:1,opt-
048:1,opt-049:1,opt-050:1"
      index="0" nodecount="33" proccount="8" reqid="0"
starttime="1162876617"></range>
  </par>
  <par Name="32Bit">
    <range duration="1200" hostlist="x86-001:1,x86-002:1,x86-004:1,x86-006:1,x86-
013:1,x86-014:1,x86-015:1,x86-016:1,x86-037:1"
      index="0" nodecount="9" proccount="8" reqid="0"
starttime="1162425519"></range>
    <range duration="1200" hostlist="x86-001:1,x86-002:1,x86-004:1,x86-006:1,x86-
013:1,x86-014:1,x86-015:1,x86-016:1,x86-037:1,x86-042:1,x86-043:1"
      index="0" nodecount="11" proccount="8" reqid="0"

```

```

starttime="1162956803"></range>
  <range duration="1200" hostlist="x86-001:1,x86-002:1,x86-004:1,x86-006:1,x86-
013:1,x86-014:1,x86-015:1,x86-016:1,x86-027:1,x86-028:1,
  x86-029:1,x86-030:1,x86-037:1,x86-041:1,x86-042:1,x86-043:1,x86-046:1,x86-
047:1,x86-048:1,x86-049:1"
  index="0" nodecount="20" proccount="8" reqid="0"
starttime="1163053393"></range>
</par>
</Data>

```

**i** This command reports the original query, and the timeframe, resource size, and hostlist associated with each possible time slot.

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mshow in a hosting environment](#)

### mshow -a

## Basic Current and Future Requests

The `mshow` command can report information on many aspects of the scheduling environment. To request information on available resources, the `-a` flag should be used. By default, the `mshow` command resource availability query only reports resources that are immediately available. To request information on specific resources, the type of resources required can be specified using the `-w` flag as in the following example:

```
> mshow -a -w taskmem=1500,duration=600
...
```

To view current and future resource availability, the `future` flag should be set as in the following example:

```
> mshow -a -w taskmem=1500,duration=600 --flags=future
...
```

## Co-allocation Resources Queries

In many cases, a particular request will need simultaneous access to resources of different types. The `mshow` command supports a co-allocation request specified by using multiple `-w` arguments. For example, to request 16 nodes with feature `fastcpu` and 2 nodes with feature `fastio`, the following request might be used:

```
> mshow -a -w minprocs=16,duration=1:00:00,nodefeature=fastcpu -w
minprocs=2,nodefeature=fastio,duration=1:00:00 --flags=future
Partition      Procs   Nodes   StartOffset      Duration      StartDate      ReqID=0
-----
ALL            16      8       00:00:00         1:00:00      13:00:18_08/25
ALL             2       1       00:00:00         1:00:00      13:00:18_08/25      ReqID=1
```

The [mshow -a](#) documentation contains a list of the different resources that may be queried as well as examples on using `mshow`.

## Using Transaction IDs

By default, the `mshow` command reports simply when and where the requested resources are available. However, when the `tid` flag is specified, the `mshow` command returns both resource availability information and a handle to these resources called a Transaction ID as in the following example:

```
> mshow -a -w minprocs=16,nodefeature=fastcpu,duration=2:00:00 --flags=future,tid
Partition      Procs  Nodes  StartOffset      Duration      StartDate
-----
ALL            16    16      00:00:00        2:00:00      13:00:18_08/25  TID=26 ReqID=0
```

In the preceding example, the returned transaction id (TID) may then be used to reserve the available resources using the [mrsvctl -c -R](#) command:

```
> mrsvctl -c -R 26
reservation system.1 successfully created
```

Any TID can be printed out using the [mschedctl -l trans](#) command:

```
Code example (replace with your own content)
> mschedctl -l trans 26 TID[26] A1='node01' A2='600' A3='1093465728' A4='ADVRES' A5='fastio'
```

Where A1 is the hostlist, A2 is the duration, A3 is the starttime, A4 are any flags, and A5 are any features.

## Using Reservation Profiles

Reservation profiles ([RSVPROFILE](#)) stand as templates against which reservations can be created. They can contain a hostlist, starttime, endtime, duration, access-control list, flags, triggers, variables, and most other attributes of an Administrative Reservation. The following example illustrates how to create a reservation with the exact same trigger-set.

```

-----
# moab.cfg
-----
RSVPROFILE[test1] TRIGGER=Sets=$Var1.$Var2.$Var3.!Net,EType=start,AType=exec,
    Action=/tmp/host/triggers/Net.sh,
    Timeout=1:00:00
RSVPROFILE[test1]                                TRIGGER=Requires=$Var1.$Var2.$Var3,
    Sets=$Var4.$Var5,EType=start,
    AType=exec,Action=/tmp/host/triggers/
    FS.sh+$Var1:$Var2:$Var3,Timeout=20:00
RSVPROFILE[test1]
TRIGGER=Requires=$Var1.$Var2.$Var3.$Var4.$Var5,
    Sets=!NOOSinit.OSinit,EType=start,
    AType=exec,
    Action=/tmp/host/triggers/
    OS.sh+$Var1:$Var2:$Var3:$Var4:$Var5
RSVPROFILE[test1]
TRIGGER=Requires=NOOSini,AType=cancel,EType=start
RSVPROFILE[test1]
TRIGGER=EType=start,Requires=OSinit,AType=exec,
    Action=/tmp/host/triggers/success.sh
...
-----

```

To create a reservation with this profile the `mrsvctl -c -P` command is used:

```

> mrsvctl -c -P test1
reservation system.1 successfully created

```

## Using Reservation Groups

Reservation groups are a way for Moab to tie reservations together. When a reservation is created using multiple Transaction IDs, these transactions and their resulting reservations are tied together into one group.

```

> mrsvctl -c -R 34,35,36
reservation system.99 successfully created
reservation system.100 successfully created
reservation system.101 successfully created

```

In the preceding example, these three reservations would be tied together into a single group. The `mdiag -r` command can be used to see which group a reservation belongs to. The `mrsvctl -q diag -g` command can also be used to print out a specific group of reservations. The `mrsvctl -c -g` command can also be used to release a group of reservations.

### Related topics

- [mshow](#)

# msub

## Synopsis

```
msub [-a datetime][-A account][-c interval][-C directive_prefix][-d path] [-e path][-E][-F][-h][-l][-j join][-k keep][-K][-l resourcelist][-m mailoptions] [-M user_list][-N name][-o path][-p priority][-q destination][-r] [-S pathlist][-t jobarrays][-u userlist][-v variablelist][-V] [-W additionalattributes][-x][-z][--stagein][--stageout][--stageinfile][--stageoutfile][--stageinsize][--stageoutsize][--workflowjobsids][script]
```

## Overview

msub allows users to submit jobs directly to Moab. When a job is submitted directly to a resource manager (such as TORQUE), it is constrained to run on only those nodes that the resource manager is directly monitoring. In many instances, a site may be controlling multiple resource managers. When a job is submitted to Moab rather than to a specific resource manager, it is not constrained as to what nodes it is executed on. msub can accept command line arguments (with the same syntax as qsub), job scripts (in either PBS or LoadLeveler syntax), or the SSS Job XML specification.

**i** Moab must run as a root user in order for msub submissions to work. Workload submitted via msub when Moab is running as a non-root user fail immediately.

Submitted jobs can then be viewed and controlled via the [mjobctl](#) command.

**i** Flags specified in the following table are not necessarily supported by all resource managers.

## Access

When Moab is configured to run as root, any user may submit jobs via msub.

## Flags

-a	
<b>Name</b>	Eligible Date
<b>Format</b>	[[[[[CC]YY]MM]DD]hhmm[.SS]
<b>Description</b>	Declares the time after which the job is eligible for execution.
<b>Example</b>	<pre>&gt; msub -a 12041300 cmd.pbs</pre> <p><i>Moab will not schedule the job until 1:00 pm on December 4, of the current year.</i></p>

-A	
<b>Name</b>	Account
<b>Format</b>	<ACCOUNT NAME>
<b>Description</b>	Defines the account associated with the job.
<b>Example</b>	<pre>&gt; msub -A research cmd.pbs</pre> <p><i>Moab will associate this job with account research.</i></p>

-c	
<b>Name</b>	Checkpoint Interval
<b>Format</b>	[n s c c=<minutes>]
<b>Description</b>	<p>Checkpoint of the will occur at the specified interval.</p> <p><b>n</b> — No Checkpoint is to be performed.  <b>s</b> — Checkpointing is to be performed only when the server executing the job is shut down.  <b>c</b> — Checkpoint is to be performed at the default minimum time for the server executing the job.  <b>c=&lt;minutes&gt;</b> — Checkpoint is to be performed at an interval of minutes.</p>
<b>Example</b>	<pre>&gt; msub -c c=12 cmd.pbs</pre> <p><i>The job will be checkpointed every 12 minutes.</i></p>

-C	
<b>Name</b>	Directive Prefix
<b>Format</b>	'<PREFIX NAME>'
<b>Default</b>	First known prefix (#PBS, #@, #BSUB, #!, #MOAB, #MSUB)

-C	
<b>Description</b>	<p>Specifies which directive prefix should be used from a job script.</p> <ul style="list-style-type: none"> <li>• It is best to submit with single quotes. '#PBS'</li> <li>• An empty prefix will cause Moab to not search for any prefix. -C ''</li> <li>• Command line arguments have precedence over script arguments.</li> <li>• Custom prefixes can be used with the -C flag. -C '#MYPREFIX'</li> <li>• Custom directive prefixes must use PBS syntax.</li> <li>• If the -C flag is not given, Moab will take the first default prefix found. Once a directive is found, others are ignored.</li> </ul>
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; msub -C '#MYPREFIX' cmd.pbs #MYPREFIX -l walltime=5:00:00 (in cmd.pbs)</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Moab will use the #MYPREFIX directive specified in cmd.pbs, setting the wallclock limit to five hours.</i></p>

-d	
<b>Name</b>	Execution Directory
<b>Format</b>	<path>
<b>Default</b>	Depends on the RM being used. If using TORQUE, the default is \$HOME. If using SLURM, the default is the submission directory.
<b>Description</b>	Specifies which directory the job should execute in.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; msub -d /home/test/job12 cmd.pbs</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>The job will begin execution in the /home/test/job12 directory.</i></p>

-e	
<b>Name</b>	Error Path
<b>Format</b>	[<hostname>:]<path>
<b>Default</b>	\$SUBMISSIONDIR/\$JOBNAME.e\$JOBID

-e	
<b>Description</b>	Defines the path to be used for the standard error stream of the batch job.
<b>Example</b>	<pre>&gt; msub -e test12/stderr.txt</pre> <p><i>The STDERR stream of the job will be placed in the relative (to execution) directory specified.</i></p>

-E	
<b>Name</b>	Environment Variables
<b>Description</b>	<p>Moab adds the following variables, if populated, to the job's environment:</p> <ul style="list-style-type: none"> <li>• MOAB_ACCOUNT — Account name.</li> <li>• MOAB_BATCH — Set if a batch job (non-interactive).</li> <li>• MOAB_CLASS — Class name.</li> <li>• MOAB_DEPEND — Job dependency string.</li> <li>• MOAB_GROUP — Group name.</li> <li>• MOAB_JOBARRAYINDEX — For a job in an array, the index of the job.</li> <li>• MOAB_JOBARRAYRANGE — For a system with job arrays, the range of all job arrays.</li> <li>• MOAB_JOBID — Job ID. If submitted from the grid, grid jobid.</li> <li>• MOAB_JOBNAME — Job name.</li> <li>• MOAB_MACHINE — Name of the machine (i.e. Destination RM) that the job is running on.</li> <li>• MOAB_NODECOUNT — Number of nodes allocated to job.</li> <li>• MOAB_NODELIST — Comma-separated list of nodes (listed singly with no ppn info).</li> <li>• MOAB_PARTITION — Partition name the job is running in. If grid job, cluster scheduler's name.</li> <li>• MOAB_PROCCOUNT — Number of processors allocated to job.</li> <li>• MOAB_QOS — QOS name.</li> <li>• MOAB_TASKMAP — Node list with procs per node listed. &lt;nodename&gt;.&lt;procs&gt;</li> <li>• MOAB_USER — User name.</li> </ul> <p>In SLURM environments, not all variables will be populated since the variables are added at submission (such as NODELIST). With TORQUE/PBS, the variables are added just before the job is started.</p> <p>This feature only works with SLURM and TORQUE/PBS.</p>
<b>Example:</b>	<pre>&gt; msub -E mySim.cmd</pre> <p><i>The job mySim will be submitted with extra environment variables.</i></p>

-F	
<b>Name</b>	Script Flags
<b>Format</b>	"\"<STRING>\""
<b>Description</b>	<p>Specifies the flags TORQUE will pass to the job script at execution time.</p> <div style="border: 1px solid #4a6982; border-radius: 10px; padding: 5px; background-color: #d9e1f2;"> <p> The <code>-F</code> flag is only compatible with TORQUE resource managers.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; border-radius: 10px; padding: 10px; background-color: #f0f0f0;"> <pre>&gt; msub -F "\"arg1 arg2\"" -l nodes=1,walltime=60 files/job.sh</pre> <p><i>TORQUE will pass parameters <code>arg1</code> and <code>arg2</code> to the <code>job.sh</code> script when the job executes.</i></p> </div>

-h	
<b>Name</b>	Hold
<b>Description</b>	Specifies that a user hold be applied to the job at submission time.
<b>Example</b>	<div style="border: 1px dashed #ccc; border-radius: 10px; padding: 10px; background-color: #f0f0f0;"> <pre>&gt; msub -h cmd.ll</pre> <p><i>The job will be submitted with a user hold on it.</i></p> </div>

-I	
<b>Name</b>	Interactive
<b>Description</b>	<p>Declares the job is to be run interactively.</p> <div style="border: 1px solid #4a6982; border-radius: 10px; padding: 5px; background-color: #d9e1f2;"> <p> <code>qsub</code> must exist on the same host as <code>msub</code> if the interactive job is destined for a TORQUE cluster, because the interactive <code>msub</code> request will be converted to a <code>qsub -l</code> request.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; border-radius: 10px; padding: 10px; background-color: #f0f0f0;"> <pre>&gt; msub -I job117.sh</pre> <p><i>The job will be submitted in interactive mode.</i></p> </div>

-j	
<b>Name</b>	Join
<b>Format</b>	[eo oe n]
<b>Default</b>	n (not merged)
<b>Description</b>	<p>If <code>eo</code> is specified, the error and output streams are merged into the <i>error</i> stream. If <code>oe</code> is specified, the error and output streams will be merged into the <i>output</i> stream.</p> <div style="border: 1px solid #004a7c; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p><b>i</b> If using either the <code>-e</code> or the <code>-o</code> option and the <code>-j eo oe</code> option, the <code>-j</code> option takes precedence and all standard error and output messages go to the chosen output file.</p> </div>
<b>Example</b>	<pre>&gt; msub -j oe cmd.sh</pre> <p><i>STDOUT and STDERR will be merged into one file.</i></p>

-k	
<b>Name</b>	Keep
<b>Format</b>	[e o eo oe n]
<b>Default</b>	n (not retained)
<b>Description</b>	Defines which (if either) of output and error streams will be retained on the execution host (overrides path for stream).
<b>Example</b>	<pre>&gt; msub -k oe myjob.sh</pre> <p><i>STDOUT and STDERR for the job will be retained on the execution host.</i></p>

-K	
<b>Name</b>	Continue Running
<b>Format</b>	N/A

-K	
<b>Description</b>	<p>Tells the client to continue running until the submitted job is completed. The client will query the status of the job every 5 seconds. The time interval between queries can be specified or disabled via <a href="#">MSUBQUERYINTERVAL</a>.</p> <div style="border: 1px solid #4a7c9c; background-color: #e6f2ff; padding: 5px; margin-top: 10px;"> <p><b>i</b> Use the <code>-K</code> option sparingly (if at all) as it slows down the Moab scheduler with frequent queries. Running ten jobs with the <code>-K</code> option creates an additional fifty queries per minute for the scheduler.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 10px;">&gt; msub -K newjob.sh 3 Job 3 completed*</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"><i>*Only shows up after job completion.</i></p>

-l	
<b>Name</b>	Resource List
<b>Format</b>	<p>&lt;STRING&gt;                      -l [BANDWIDTH DDISK DEADLINE DEPEND DMEM EXCLUDENODES FEATURE...[]                      Additional options can be referenced on the <a href="#">resource manager extensions</a> page.</p>
<b>Description</b>	<p>Defines the resources that are required by the job and establishes a limit to the amount of resource that can be consumed. Resources native to the resource manager, scheduler <a href="#">resource manager extensions</a>, or <a href="#">job flags</a> may be specified. Note that resource lists are dependent on the resource manager in use.</p> <p>For information on specifying multiple types of resources for allocation, see <a href="#">Multi-Req Support</a>.</p>

**-l****Example**

```
> msub -l nodes=32:ppn=2,pmem=1800mb,walltime=3600,VAR=testvar:myvalue cmd.sh
> msub -l nodes=32:ppn=2,pmem=1800mb,walltime=3600,VAR=testvar:
myvalue cmd.sh
```

*The job requires 32 nodes with 2 processors each, 1800 MB per task, a walltime of 3600 seconds, and a variable named `testvar` with a value of `myvalue`.*

**i** If `JOBNODEMATCHPOLICY` is not set, Moab does not reserve the requested number of processors on the requested number of nodes. It reserves the total number of requested processors (`nodes x ppn`) on any number of nodes. Rather than setting `nodes=<value>:ppn=<value>`, set `procs=<value>`, replacing `<value>` with the total number of processors the job requires. Note that `JOBNODEMATCHPOLICY` is not set by default.

```
> msub -l nodes=32:ppn=2 -l advres=!<resvid>
```

*This entry would tell Moab to only consider resources outside of the specified `<reservation id>`.*

**-m****Name**

Mail Options

**Format**<STRING> (either `n` or one or more of the characters `a`, `b`, and `e`)**Description**Defines the set of conditions (`abort`,`begin`,`end`) when the server will send a mail message about the job to the user.**Example**

```
> msub -m be cmd.sh
```

*Mail notifications will be sent when the job begins and ends.*

**-M****Name**

Mail List

**Format**

&lt;user&gt;[@&lt;host&gt;][,&lt;user&gt;[@&lt;host&gt;],...]

**Default**

\${JOBOWNER}

-M	
<b>Description</b>	Specifies the list of users to whom mail is sent by the execution server. Overrides the <b>EMAILADDRESS</b> specified on the <a href="#">USERCFG</a> credential.
<b>Example</b>	<pre>&gt; msub -M jon@node01,bill@node01,jill@node02 cmd.sh</pre> <p><i>Mail will be sent to the specified users if the job is aborted.</i></p>

-N	
<b>Name</b>	Name
<b>Format</b>	<STRING>
<b>Default</b>	STDIN or name of job script
<b>Description</b>	Specifies the user-specified job name attribute.
<b>Example</b>	<pre>&gt; msub -N chemjob3 cmd.sh</pre> <p><i>Job will be associated with the name chemjob3.</i></p>

-O	
<b>Name</b>	Output Path
<b>Format</b>	[<hostname>:]<path> - %J and %I are acceptable variables. %J is the master array name and %I is the array member index in the array.
<b>Default</b>	\$SUBMISSIONDIR/\$JOBNAME. o\$JOBID

-o	
<b>Description:</b>	<p>Defines the path to be used for the standard output stream of the batch job.</p> <p>More variables are allowed when they are used in the job script instead of <code>msub -o</code>. In the job script, specify a <code>#PBS -o</code> line and input your desired variables. The allowable variables are:</p> <ul style="list-style-type: none"> <li>• OID</li> <li>• OTYPE</li> <li>• USER</li> <li>• OWNER</li> <li>• JOBID</li> <li>• JOBNAME</li> </ul> <p>Submitting a job script that has the line <code>#PBS -o \$(USER)_\$(JOBID)_\$(JOBNAME).txt</code> results in a file called <code>&lt;username&gt;_&lt;jobID&gt;_&lt;jobName&gt;.txt</code>.</p> <p>Do not use <code>msub -o</code> when submitting a job script that has a <code>#PBS -o</code> line defined.</p>
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; msub -o test12/stdout.txt</pre> <p style="text-align: center;"><i>The STDOUT stream of the job will be placed in the relative (to execution) directory specified.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px;"> <pre>&gt; msub -t 1-2 -o /home/jsmith/simulations/%J-%I.out ~/sim5.sh</pre> <p style="text-align: center;"><i>A job array is submitted and the name of the output files includes the master array index and the array member index.</i></p> </div>

-p	
<b>Name</b>	Priority
<b>Format</b>	<code>&lt;INTEGER&gt;</code> (between -1024 and 0)
<b>Default</b>	0
<b>Description</b>	<p>Defines the priority of the job.</p> <p>To enable priority range from -1024 to +1023, see <a href="#">ENABLEPOSUSERPRIORITY</a>.</p>
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; msub -p 25 cmd.sh</pre> <p style="text-align: center;"><i>The job will have a user priority of 25.</i></p> </div>

-q	
<b>Name</b>	Destination Queue (Class)
<b>Format</b>	[<queue>][@<server>]
<b>Default</b>	[<DEFAULT>]
<b>Description</b>	Defines the destination of the job.
<b>Example</b>	<pre>&gt; msub -q priority cmd.sh</pre> <p><i>The job will be submitted to the priority queue.</i></p>

-r	
<b>Name</b>	Rerunable
<b>Format</b>	[y n]
<b>Default</b>	n
<b>Description:</b>	Declares whether the job is rerunable.
<b>Example</b>	<pre>&gt; msub -r n cmd.sh</pre> <p><i>The job cannot be rerun.</i></p>

-S	
<b>Name</b>	Shell
<b>Format</b>	<path>[@<host>][,<path>[@<host>],...]
<b>Default</b>	\$SHELL
<b>Description</b>	Declares the shell that interprets the job script.

**-S**

<b>Example</b>	<pre>&gt; msub -S /bin/bash</pre> <p><i>The job script will be interpreted by the /bin/bash shell.</i></p>
----------------	------------------------------------------------------------------------------------------------------------

**-t**

<b>Name</b>	Job Arrays
<b>Format</b>	<name>.[<indexlist>]%<limit>
<b>Description</b>	Starts a job array with the jobs in the index list. The limit variable specifies how many jobs may run at a time. For more information, see <a href="#">Submitting Job Arrays</a> .
	<div style="border: 1px solid #004a80; padding: 5px;"> <p><b>i</b> Moab enforces an internal limit of 100,000 sub-jobs that a single array job submission can specify.</p> </div>
<b>Example</b>	<pre>&gt; msub -t myarray.[1-1000]%4</pre>

**-u**

<b>Name</b>	User List
<b>Format</b>	<user>[@<host>[,<user>[@<host>],...]]
<b>Default</b>	UID of msub command
<b>Description</b>	Defines the user name under which the job is to run on the execution system.
<b>Example</b>	<pre>&gt; msub -u bill@node01 cmd.sh</pre> <p><i>On node01 the job will run under Bill's UID, if permitted.</i></p>

**-v**

<b>Name</b>	Variable List
-------------	---------------

-v	
<b>Format</b>	<string>[,<string>,...]
<b>Description</b>	Expands the list the environment variables that are exported to the job (taken from the msub command environment).
<b>Example</b>	<pre>&gt; msub -v DEBUG cmd.sh</pre> <p><i>The DEBUG environment variable will be defined for the job.</i></p>

-V	
<b>Name</b>	All Variables
<b>Description</b>	Declares that all environment variables in the msub environment are exported to the batch job
<b>Example</b>	<pre>&gt; msub -V cmd.sh</pre> <p><i>All environment variables will be exported to the job.</i></p>

-W	
<b>Name</b>	Additional Attributes
<b>Format</b>	<string>
<b>Description</b>	Allows for specification of additional job attributes (See <a href="#">Resource Manager Extension</a> )
<b>Example</b>	<pre>&gt; msub -W x=GRES:matlab:1 cmd.sh</pre> <p><i>The job requires one resource of matlab.</i></p> <p>This flag can be used to set a filter for what namespaces will be passed from a job to a trigger using a comma-delimited list. This limits the trigger's action to objects contained in certain workflows. For more information, see <a href="#">Requesting name space variables on page 778</a>.</p> <pre>&gt; msub -W x="trigns=vc1,vc2"</pre> <p><i>The job passes namespaces vc1 and vc2 to triggers.</i></p>

-x	
<b>Format</b>	<script> or <command>
<b>Description</b>	<p>When running an interactive job, the <code>-x</code> flag makes it so that the corresponding script won't be parsed for PBS directives, but is instead a command that is launched once the interactive job has started. The job terminates at the completion of this command. This option works only when using TORQUE.</p> <div style="border: 1px solid #4a7c9c; padding: 5px; background-color: #e6f2ff;"> <p><b>i</b> The <code>-x</code> option for <code>msub</code> differs from <code>qsub</code> in that <code>qsub</code> does not require the script name to come directly after the flag. The <code>msub</code> command requires a script or command immediately after the <code>-x</code> declaration.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">&gt; msub -I -x ./script.pl &gt; msub -I -x /tmp/command</pre>

-z	
<b>Name</b>	Silent Mode
<b>Description</b>	The job's identifier will not be printed to stdout upon submission.
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">&gt; msub -z cmd.sh</pre> <div style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"> <p><i>No job identifier will be printout the stdout upon successful submission.</i></p> </div>

## Staging data

Data staging, or the ability to copy data required for a job from one location to another or to copy resulting data to a new location (See [About data staging on page 901](#) for more information), must be specified at job submission. To stage data in, you would use the `msub --stagein` and/or `--stageinfile` option, optionally with `--stageinsize`. You would use similar options the same way for staging out: `--stageout`, `--stageoutfile`, and `--stageoutsize`. `--stagein` and `--stageout`, which you can use multiple times in the same `msub` command, allow you to specify a single file or directory to stage in or out. `--stageinfile` and `--stageoutfile` allow you to specify a text file that lists the files to stage in or out. The `--stageinsize` and `[--stageoutsize]` options allow you to estimate the total size of the files and directories that you want to stage in or out, which can help Moab make an intelligent guess about how long it will take to stage the data in or out, thus ensuring that the job can start as soon as possible after the staging has occurred.

### Staging a file or directory

The `--stagein` and `--stageout` options use the same format.

```
--<stagein|stageout><=| ><source>%<destination>
```

Where *<source>* and *<destination>* take on the following format:

```
[<user>@] <host>: / <path> [ / <fileName> ]
```

Specifying a user and file name are optional. If you do not specify a file name, Moab will assume a directory.

```
> msub ... --stagein=student@biology:/stats/file001%admin@moab:/tmp/staging
<jobScript>
```

*This msub command tells Moab that the job requires file001 from student's stats directory on the biology server to be staged to admin's staging directory on the moab server prior to the job's starting.*

You can specify the option multiple times for the same `msub` command; however, staging large number of files is easier with `--stageinfile` or `--stageoutfile`.

You can also use `#MSUB` or `#PBS` within a job script to specify data staging options. For example:

```
#MSUB --stageinsize=1gb
#MSUB --stagein=...
```

See [Sample user job script on page 922](#) for more information. Note that the data staging options are not compatible with `qsub`.

### Staging multiple files or directories

The `--stageinfile` and `--stageoutfile` options use the same format. You must include the path to a text file that lists each file to stage in or out on its own line. Each file specification follows the same format as a `--stagein` or `--stageout` specification as described above. The format of the command options looks like this:

```
--<stageinfile|stageoutfile><=| ><path>/<fileName>
```

The file contains multiple lines with the following format:

```
[<user>@] <host>: / <path> [ / <fileName> ] % [ <user>@ ] <host>: / <path> [ / <fileName> ]
...
```

Moab ignores blank lines in the file. You can comment out lines by preceding them with a pound sign (`#`). The following examples demonstrate what the `--stageinfile` option looks like on the command line and what the file it specifies might look like.

```
> msub ... --stageinfile=/tmp/myStagingFile <jobScript>
```

```
/tmp/myStagingFile:
```

```
student@biology:/stats/file001%moab:/tmp/staging
student@biology:/stats/file002%moab:/tmp/staging
student@biology:/stats/file003%moab:/tmp/staging
#student@biology:/stats/file004%moab:/tmp/staging
student@biology:/stats/file005%moab:/tmp/staging

student@biology:/stats/file006%moab:/tmp/staging
student@biology:/stats/file007%moab:/tmp/staging
student@biology:/stats/file008%moab:/tmp/staging
student@biology:/stats/file009%moab:/tmp/staging
student@biology:/stats/file010%moab:/tmp/staging
```

*Moab stages in each file listed in `myStagingFile` to the `/tmp/staging` directory. Each file resides on the biology host as the student user. Moab ignores the blank line and the line specifying `file004`.*

### Stage in or out file size

The optional `--stageinsize` and `--stageoutsize` options give you the opportunity to estimate the size of the file(s) or directory(-ies) being staged to aid Moab in choosing an appropriate start time. Both options use the same format:

```
--<stageinsize|stageoutsize>=<integer>[unit]
```

The integer indicates the size of the file(s) and directory(-ies) in megabytes unless you specify a different unit. Moab accepts the follow case-insensitive suffixes: KB, MB, GB, or TB.

```
> msub --stageinfile=/stats/file003 --stageinsize=100 <jobScript>
```

*Moab copies the `/davidharris/research/recordlist` file, which is approximately 100 megabytes, from the biology node to the host where the job will run prior to job start.*

```
> msub --stageinfile=/stats/file002 --stageinsize=1gb <jobScript>
```

*Moab copies all files specified in the `/davidharris/research/recordlist` file, which add up to approximately 1 gigabyte, to the host where the job will run prior to job start.*

### Return all the job IDs in the workflow at submission time

By default, `msub` will print the job ID to stdout at the time of submission. If you want `msub` to print all of the jobs that are created as part of the workflow template, you can use the `msub --workflowjobids` option to show all the job IDs at submission time:

```
$ echo sleep 60 | msub -l walltime=15 --workflowjobids
```

```
MoabA.3.dsin MoabA.3 MoabA.3.dsout
```

### Job Script

The `msub` command supports job scripts written in any one of the following languages:

Language	Notes
<a href="#">PBS/TORQUE Job Submission Language</a>	---

Language	Notes
<a href="#">LoadLeveler Job Submission Language</a>	Use the <a href="#">INSTANTSTAGE</a> parameter as only a subset of the command file keywords are interpreted by Moab.
<a href="#">SSS XML Job Object Specification</a>	---
<b>LSF Job Submission Language</b>	enabled in Moab 4.2.4 and higher

## /etc/msubrc

Sites that wish to automatically add parameters to every job submission can populate the file `/etc/msubrc` with global parameters that every job submission will inherit.

For example, if a site wished every job to request a particular generic resource they could use the following `/etc/msubrc`:

```
-W x=GRES:matlab:2
```

## Usage Notes

`msub` is designed to be as flexible as possible, allowing users accustomed to PBS, LSF, or LoadLeveler syntax, to continue submitting jobs as they normally would. It is not recommended that different styles be mixed together in the same `msub` command.

When only one resource manager is configured inside of Moab, all jobs are immediately staged to the only resource manager available. However, when multiple resource managers are configured Moab will determine which resource manager can run the job soonest. Once this has been determined, Moab will stage the job to the resource manager.

It is possible to have Moab take a best effort approach at submission time using the `forward` flag. When this flag is specified, Moab will do a quick check and make an intelligent guess as to which resource manager can run the job soonest and then immediately stage the job.

Moab can be configured to instantly stage a job to the underlying resource manager (like TORQUE/LOADLEVELER) through the parameter [INSTANTSTAGE](#). When set inside `moab.cfg`, Moab will migrate the job instantly to an appropriate resource manager. Once migrated, Moab will destroy all knowledge of the job and refresh itself based on the information given to it from the underlying resource manager.

In most instances Moab can determine what syntax style the job belongs to (PBS or LoadLeveler); if Moab is unable to make a guess, it will default the style to whatever resource manager was configured at compile time. If LoadLeveler and PBS were both compiled then LoadLeveler takes precedence.

Moab can translate a subset of job attributes from one syntax to another. It is therefore possible to submit a PBS style job to a LoadLeveler resource manager, and vice versa, though not all job attributes will be translated.

## Examples

### Example 3-49:

```
> msub -l nodes=3:ppn=2,walltime=1:00:00,pmem=100kb script2.pbs.cmd
4364.orion
```

### Example 3-50:

This example is the XML-formatted version of the above example. See [Submitting Jobs via msub in XML](#) for more information.

```
<job>
  <InitialWorkingDirectory>/home/user/test/perlAPI
</InitialWorkingDirectory>
  <Executable>/home/user/test/perlAPI/script2.pbs.cmd
</Executable>
  <SubmitLanguage>PBS</SubmitLanguage>
  <Requested>
    <Feature>ppn2</Feature>
    <Processors>3</Processors>
    <WallclockDuration>3600</WallclockDuration>
  </Requested>
</job>
```

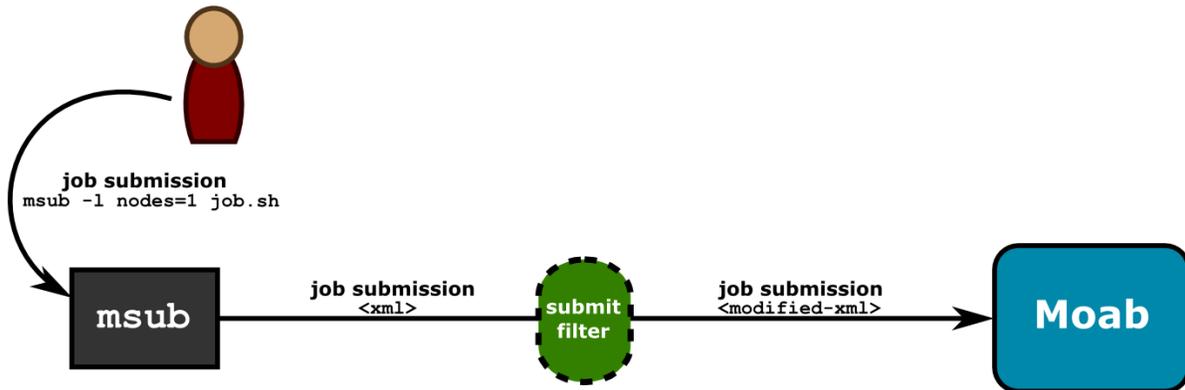
### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mjobctl](#) command to view, modify, and cancel jobs
- [checkjob](#) command to view detailed information about the job
- [mshow](#) command to view all jobs in the queue
- [DEFAULTSUBMITLANGUAGE](#) parameter
- [MSUBQUERYINTERVAL](#) parameter
- [SUBMITFILTER](#) parameter
- [Applying the msub Submit Filter](#) for job script sample

### Applying the msub submit filter

When you use [msub](#) to submit a job, `msub` processes the input, converts it to XML, and sends the job specification XML to the Moab scheduler. You can create a submission filter to modify the job XML based on the criteria you set before Moab receives and processes it.

Image 3-1: Job submission process



The filter gives you the ability to customize the submission process, which is helpful if jobs should have certain defaults assigned to them, if you want to keep detailed submission statistics, or if you want to change job requests based on custom needs.

The submit filter, is a simple executable or script that receives XML via its standard input and returns the modified XML in its standard output. It modifies the attributes of the job specification XML based on policies you specify. It can perform various other actions at your request, too; for instance, logging. Once the submit filter has modified the job XML based on your criteria, it writes the XML representing the actual job submission to stdout. The new XML could potentially match the original XML, depending on whether the job met the criteria for modification set in the job submit filter script. Job submissions you want to proceed will leave the filter with an exit code of 0 and continue to Moab for scheduling. If the job meets the filter's specified criteria for rejection, it exits with a non-zero value, aborting the job submission process. You can configure the filter script to write a descriptive rejection message to stderr.

Job submit filters follow these rejection rules: 1) `msub` will reject job XML with an exit code of anything other than zero, 2) the `msub` command displays filter's error output on the command line, 3) `msub` will reject the job if the filter outputs invalid job XML, and 4) `msub` will reject the job if it violates any policies in your general Moab configuration; you cannot use a submit filter to bypass other policies.

To see the schema for job submission XML, please refer to [Submitting Jobs via msub in XML](#).

## Submit filter types

You can implement submit filters on either the client or server side of a job submission. The primary differences between the two submit filter types are the location from which the filter runs, the powers and privileges of the user running the filter, and whether a user can bypass the filter. Client-based submit filters run from the `msub` client as the user who submits the job and can be bypassed, and server-based submit filters run from the Moab server as the user as which the server is running and cannot be bypassed.

### Client-based submit filter

Client-based filters run from the `msub` client as the user who is submitting the job. Because they do not have elevated privileges, the risk of client-based submit filters' being abused is low; however, it is

possible for the client to specify its own configuration file and bypass the filter or substitute its own filter. Job submissions do not even reach the server if a client-based submit filter rejects it.

To configure `msub` to use the submit filter, give each submission host access to the submit filter script and add a `SUBMITFILTER` parameter to the Moab configuration file (`moab.cfg`) on each submission host. The following example demonstrates how you might modify the `moab.cfg` file:

```
SUBMITFILTER /home/submitfilter/filter.pl
```

If you experience problems with your submit filter and want to debug its interaction with `msub`, enter `msub --loglevel=9`. This will cause `msub` to print verbose log messages to the terminal.

### Server-based submit filter

Server-based submit filters run from the Moab server as the user as which the server is running. Because it runs as a privileged user, you must evaluate the script closely for security implications. A client configuration cannot bypass the filter.

To configure Moab to automatically apply a filter to all job submissions, use the `SERVERSUBMITFILTER` on page 1041 parameter. `SERVERSUBMITFILTER` specifies the path to a global job submit filter script, which Moab will run on the head node and apply to every job submitted.

```
SERVERSUBMITFILTER /opt/moab/scripts/jobFilter.pl
```

*Moab runs `jobFilter.pl`, located in the `/opt/moab/scripts` directory, on the head node, applying the filter to all jobs submitted.*

### Sample submit filter script

The following example is a trivial implementation that will not affect whether a job is submitted. Use it as reference to verify that you are writing your filter properly.

```
#!/usr/bin/perl
use strict;

## Simple filter example that re-directs the output to a file.

my $file = "xmllog.out";

open FILE, ">>$file" or die "Couldn't open $file: $!";
while (<>)
{
    print FILE;
    print;
}
close FILE;
```

### Submitting Jobs via `msub` in XML

The following describes the XML format used with the `msub` command to submit a job to a Moab server. This information can be used to implement a filter and modify the XML normally generated by the `msub` command. The XML format described in what follows is based on a variant of the [Scalable Systems Software Job Object Specification](#).

## Overall XML Format

The overall format of an XML request to submit a job can be shown through the following example:

```
<job>
**job attribute children**
</job>
```

An example of a simple job element with all the required children for a job submission is as follows:

```
<job>
  <Owner>user</Owner>
  <UserId>user</UserId>
  <GroupId>group</GroupId>
  <InitialWorkingDirectory>/home/user/directory</InitialWorkingDirectory>
  <UMask>18</UMask>
  <Executable>/full/path/to/script/or/first/line/of/stdin</Executable>
  <SubmitLanguage>Resource Manager Type</SubmitLanguage>
  <SubmitString>\START\23!/usr/bin/ruby\0contents\20of\20script</SubmitString>
</job>
```

The section that follows entitled Job Element Format describes the possible attributes and their meanings in detail. In actuality, all that is needed to run a job in Moab is something similar to the following:

```
<job>
  <SubmitString>\START\23!/bin/sh\0asleep\201000</SubmitString>
</job>
```

*This piece of XML requests Moab to submit a job using the contents of the SubmitString tag as a script, which is in this case a simple sh script to sleep for 1000 seconds. The msub command will create default values for all other needed attributes.*

## Job Element Format

The job element of the submission request contains a list of children and string values inside the children that represent the attribute/value pairs for the job. The earlier section, Overall XML Format, gives an example of this format. This section explains these attributes in detail.

**Arguments** — The arguments to be passed to the program are normally specified as arguments after the first argument specifying the script to be executed.

**EligibleTime** — The minimum time after which the job is eligible. This is the equivalent of the `-a` option in `msub`. Format: [ [ [ [CC] YY] MM] DD] hhmm [ . SS]

**Environment** — The semi-colon list of environment variables that are exported to the job (taken from the `msub` command environment). The `-v msub` flag, for example, adds all the environment variables present at the time `msub` is invoked. Environment variables are delimited by the `~rs;` characters. Following is an example of the results of the `msub -v arg1=1, arg2=2` command:

```
<Environment>arg1=1~rs;arg2=2~rs;</Environment>
```

**ErrorFile** — Defines the path to be used for the standard error stream of the batch job. This is equivalent to the `-e` flag in `msub`.

**Executable** — This is normally either the name of the script to be executed, or the first line of the script if it is passed to `msub` through standard input.

**Extension** — The resource manager extension string. This can be specified via the command line in a number of ways, including the `-W x=` directive. Some other requests, such as some extensions used in the `-l` flag, are also converted to an extension string. The element has the following format:

```
<Extension>x=extension</Extension>
```

See [Using the Extension Element to Submit Triggers](#) for additional information on the extension element.

**GroupId** — The string name of the group of the user submitting the job. This will correspond to the user's primary group on the operating system.

**Hold** — Specifies that a user hold be applied to the job at submission time. This is the equivalent to the `msub` flag `-h`. It will have the form:

```
<Hold>User</Hold>
```

**InitialWorkingDirectory** — Specifies in which directory the job should begin executing. This is equivalent to the `-d` flag in the `msub` command.

```
<InitialWorkingDirectory>/home/user/directory</InitialWorkingDirectory>
```

**Interactive** — Specifies that the job is to be interactive. This is the equivalent of the `-I` flag in `msub`.

```
<Interactive>TRUE</Interactive>
```

**JobName** — Specifies the user-specified job name attribute. This is equivalent to the `-N` flag in `msub`.

**NotificationList** — Specifies the job states after which an email should be sent and also specifies the users to be emailed. This is the equivalent of the `-m` and `-M` options in `msub`.

```
<NotificationList URI=user1:user2>JobFail,JobStart,JobEnd</NotificationList>
```

*In this example, the command `msub -m abc -M user1:user2` ran indicating that emails should be sent when a job fails, starts, or ends, and that they should be sent to `user1` and `user2`.*

**OutputFile** — Defines the path to be used for the standard output stream of the batch job. This is the equivalent of the `-o` flag in `msub`.

**Priority** — A user-requested priority value. This is the equivalent to the `msub -p` flag.

**ProjectId** — Defines the account associated with the job. This is equivalent to the `-A` `msub` flag.

**QueueName** — The requested class of the job. This is the equivalent of the `msub -q` flag.

**Requested** — Specifies resources and attributes the job specifically requests and has the following form:

```
<Requested>
  <... requested attributes>
</Requested>
```

See the section dedicated to requestable attributes in this element.

**RMFlags** — Flags that will get passed directly to the resource manager on job submission. This is equivalent to any arguments listed after the `-l msub` flag.

```
<RMFlags>arg1 arg2 arg3</RMFlags>
```

**ShellName** — Declares the shell that interprets the job script. This is equivalent to the `msub` flag `-S`.

**SubmitLanguage** — Resource manager whose language the job is using. Use TORQUE to specify a TORQUE resource manager.

**SubmitString** — Contains the contents of the script to be run, retrieved either from an actual script or from standard input. This also includes all resource manager specific directives that may have been in the script already or added as a result of other command line arguments.

**TaskGroup** — Groups a set of requested resources together. It does so by encapsulating a Requested element. For example, the command `msub -l nodes=2+nodes=3:ppn=2` generates the following XML:

```
<TaskGroup>
  <Requested>
    <Processors>2</Processors>
    <TPN>2</TPN>
  </Requested>
</TaskGroup>
<TaskGroup>
  <Requested>
    <Processors>2</Processors>
  </Requested>
</TaskGroup>
```

**UserId** — The string value of the user ID of the job owner. This will correspond to the user's name on the operating system.

## Using the Extension Element to Submit Triggers

Use the Extension element to submit triggers. With the exception of certain characters, the syntax for [trigger creation](#) is the same for non-XML trigger submission. See [About object triggers on page 745](#) for detailed information on triggers. The ampersand (&) and less than sign (<) characters must be replaced for the XML to be valid. The following example shows how the Extension element is used to submit multiple triggers (separated by a semi-colon). Note that ampersand characters are replaced with `&amp;` in the example:

```
<Job>
  <UserId>user1</UserId>
  <GroupId>user1</GroupId>
  <Arguments>60</Arguments>
  <Executable>/bin/sleep</Executable>

  <Extension>x=trig:AType=exec&amp;Action="env"&amp;EType=start;trig:AType=exec&amp;Action="trig2.sh"&amp;EType=end</Extension>
  <Processors>3</Processors>
  <Disk>500</Disk>
  <Memory>1024</Memory>
  <Swap>600</Swap>
  <WallclockDuration>300</WallclockDuration>
  <Environment>PERL5LIB=/perl5:</Environment>
</Job>
```

## Elements Found in Requested Element

The following describes the tags that can be found in the Requested sub-element of the job element in a job submission request.

**Nodes** — A list of nodes that the job requests to be run on. This is the equivalent of the `-l hosts=<host-list> msub` directive.

```
<Requested>
  <Nodes>
    <Node>n1:n2</Node>
  </Nodes>
</Requested>
```

*In this example, the users requested the hosts n1 and n2 with the command `msub -l host=n1:n2`.*

**Processors** — The number of processors requested by the job. The following example was generated with the command `msub -l nodes=5`:

```
<Requested>
  <Processors>5</Processors>
</Requested>
```

**TPN** — Tasks per node. This is generated using the ppn resource manager extensions. For example, from `msub -l nodes=3:ppn=2`, the following results:

```
<Requested>
  <Processors>6</Processors>
  <TPN>2</TPN>
</Requested>
```

**WallclockDuration** — The requested wallclock duration of the job. This attribute is specified in the Requested element.

```
<Requested>
  <WallclockDuration>3600</WallclockDuration>
</Requested>
```

### Related topics

- [Applying the msub Submit Filter](#)
- [SUBMITFILTER](#) parameter

## mvctl (Moab Virtual Container Control)

### Synopsis

- `mvctl -a <OType>:<OName>[,<OType>:<OName>] <name>`
- `mvctl -c [<description>]`
- `mvctl -d <name>`
- `mvctl -m <ATTR>=VAL[,<ATTR>=VAL] <name>`

- `mvctl -q` [*<name>*|ALL] [--xml][--blocking][--flags=fullxml]
- `mvctl -r` *<OType>*:*<OName>*[,*<OType>*:*<OName>*] *<name>*
- `mvctl -x` *<action>**<name>*

## Overview

A virtual container (VC) is a logical grouping of objects with a shared variable space and applied policies. Containers can hold virtual machines, jobs, reservations, and nodes. Containers can also be nested inside other containers.

A VC can be owned by a user, group, or account. Users can only view VCs to which they have access. Level 1 administrators (Admin1) can view and modify all VCs. The owner can also be changed. When modifying the owner, you must also specify the owner type:

```
mvctl -m OWNER=acct:bob myvc
```

Adding objects to VCs at submission: You associate jobs, VMs, and reservations with a specified VC upon submission. For example,

- `mrsvctl -c ... -H <VC>`
- `msub ... -W x="vc=<VC>"`
- `mvmctl -c ...,vc=<VC>`



The user who submits objects must have access to the VC or the command is rejected.

## FullXML flag

The FullXML flag will cause the `mvctl -q` command to show VCs in a hierarchical manner. If doing a non-XML (plaintext) query, sub-VCs will be listed inside their parent VCs. Each VC will be indented more than its parent.

```
VC[vc2] (vc2)
  Owner: user:jason
  VCs:
  VC[vc1] (vc1)
    Owner: user:jason
    Jobs: Moab.1
    Rsvs: system.1
    VCs:
    VC[vc3] (vc3)
      Owner: user:jason
  VC[vc4] (vc4)
    Owner: user:jason
```

If doing an XML query, the XML for all sub-objects (VCs, but also reservations, jobs, etc.) will also be included in the VC.

```

<Data>
  <vc DESCRIPTION="vc2" NAME="vc2" OWNER="user:jason">
    <vc DESCRIPTION="vc1" NAME="vc1" OWNER="user:jason">
      <job CmdFile="sleep 7200" Flags="GLOBALQUEUE,NORMSTART"
        Group="jason" JobID="Moab.1" PAL="[base]" RM="internal"
        ReqAWDDuration="2:00:00" User="jason">
        <req Index="0"></req>
      </job>
      <rsv ACL="RSV=%=system.1=;" AUser="jason"
        AllocNodeList="n0,n1,n2,n3,n4,n5,n6,n7,n8,n9" HostExp="ALL"
        HostExpIsSpecified="TRUE" Name="system.1" Partition="base"
        ReqNodeList="n0:1,n1:1,n2:1,n3:1,n4:1,n5:1,n6:1,n7:1,n8:1,n9:1"
        Resources="PROCS=[ALL]" StatCIPS="5964.00" SubType="Other"
        Type="User" ctime="1299953557" duration="3600"
        endtime="1299957157"
        flags="ISCLOSED,ISGLOBAL,ISACTIVE,REQFULL"
        starttime="1299953557">
        <ACL aff="neutral" cmp="%" name="system.1" type="RSV">
        </ACL>
        <CL aff="neutral" cmp="%" name="system.1" type="RSV"></CL>
        <History>
          <event state="PROCS=40" time="1299953557"></event>
        </History>
      </rsv>
    </vc DESCRIPTION="vc3" NAME="vc3" OWNER="user:jason"></vc>
  </vc>
  <vc DESCRIPTION="vc4" NAME="vc4" OWNER="user:jason"></vc>
</vc>
</Data>

```

*Note that the XML from the blocking and non-blocking commands may differ.*

## Virtual Container Flags

The following table indicates available virtual container (VC) flags and associated descriptions. Note that the `Deleting`, `HasStarted`, and `Workflow` flags cannot be set by a user but are helpful indicators of status.

VC Flags	
<b>DestroyObjects</b>	When the VC is destroyed, any reservations, jobs, and VMs in the VC are also destroyed. This is recursive, so any objects in sub-VCs are also destroyed. Nodes are not removed.
<b>DestroyWhenEmpty</b>	When the VC is empty, it is destroyed.
<b>Deleting</b>	Set by the scheduler when the VC has been instructed to be removed.

 Internal flag. Administrators cannot set or clear this flag.

VC Flags	
<b>HasStarted</b>	<p>This flag is set on a VC workflow where at least one job has started.</p> <div style="border: 1px solid #004a87; border-radius: 5px; padding: 5px; background-color: #e6f2ff;"> <span style="font-size: 1.2em; font-weight: bold; color: #004a87;">i</span> Internal flag. Administrators cannot set or clear this flag.                 </div>
<b>HoldJobs</b>	<p>This flag will place a hold on any job that is submitted to the VC while this flag is set. It is not applied for already existing jobs that are added into the VC. If a job with a workflow is submitted to the VC, all jobs within the workflow are placed on hold.</p>
<b>NoReleaseWhenScheduled</b>	<p>Prevents Moab from lifting the UserHold on the workflow when it is scheduled. This enables an approval method in which an administrator must release the hold manually before the service is allowed to start as scheduled.</p>
<b>Workflow</b>	<p>Designates this VC as a VC that is for workflows. This flag is set when generated by a job template workflow. Workflow jobs can only be attached to one workflow VC.</p> <div style="border: 1px solid #004a87; border-radius: 5px; padding: 5px; background-color: #e6f2ff;"> <span style="font-size: 1.2em; font-weight: bold; color: #004a87;">i</span> Internal flag. Administrators cannot set or clear this flag.                 </div>

## Format

-a	
<b>Format</b>	<pre>mvctl -a&lt;OType&gt;:&lt;OName&gt;[,&lt;OType&gt;:&lt;OName&gt;] &lt;name&gt;</pre> <p>Where &lt;OType&gt; is one of JOB, RSV, NODE, VC, or VM.</p>
<b>Description</b>	Add the given object(s).
<b>Example</b>	<pre>mvctl -a JOB:Moab.45 vc13 &gt;&gt;job 'Moab.45' added to VC 'vc13'</pre>

-c	
<b>Format</b>	<pre>mvctl -c [&lt;description&gt;]</pre>
<b>Description</b>	Create a virtual container (VC). The VC name is auto-generated. It is recommended that you supply a description; otherwise the description is the same as the auto-generated name.
<b>Example</b>	<pre>mvctl -c "Linux testing machine" &gt;&gt;VC 'vc13' created</pre>

-d	
<b>Format</b>	<code>mvccctl -d&lt;lab01&gt;</code>
<b>Description</b>	Destroy the VC.
<b>Example</b>	<pre>mvccctl -d vc13 &gt;&gt;VC 'vc13' destroyed</pre>

-m	
<b>Format</b>	<code>mvccctl -m&lt;ATTR&gt;=VAL[,&lt;ATTR&gt;=&lt;VAL&gt;] &lt;name&gt;</code>
<b>Description</b>	Modify the VC. Attributes are flags, owner, reqstarttime, reqnodeset, variables, and owner; note that only the owner can modify owner. Use reqstarttime when implementing guaranteed start time to specify when jobs should start. The reqnodeset attribute indicates the node set that jobs should run in that are submitted to a virtual container.
<b>Example</b>	<pre>mvccctl -m variables+=HV=node8 vc13 &gt;&gt;VC 'vc13' successfully modified  mvccctl -m flags+=DESTROYWHENEMPTY vc1 &gt;&gt;VC 'vc1' successfully modified</pre>

-q	
<b>Format</b>	<code>mvccctl -q [&lt;name&gt; ALL] [--xml][--blocking][--flags=fullxml]</code>
<b>Description</b>	Query VCs
<b>Example</b>	<pre>mvccctl -q ALL VC[vc13] (Linux testing machine) Create Time: 1311027343   Creator: jdoe Owner: user:jdoe ACL:  USER=%=jdoe+; Jobs: Moab.45 Vars: HV=node88 Flags: DESTROYWHENEMPTY</pre>

-r	
<b>Format</b>	<code>mvccctl -r&lt;OType&gt;:&lt;OName&gt;[,&lt;OType&gt;:&lt;OName&gt;] &lt;name&gt;</code> Where <OType> is one of JOB, RSV, NODE, VC, or VM.
<b>Description</b>	Remove the given object(s) from the VC.
<b>Example</b>	<pre>mvccctl -r JOB:Moab.45 vc13 &gt;&gt;job 'Moab.45' removed from VC 'vc13'</pre>

-x	
<b>Format</b>	<code>mvccctl -x&lt;action&gt;&lt;name&gt;</code>
<b>Description</b>	Executes the given action on the virtual container (VC).
<b>Example</b>	<pre>mvccctl -x schedulevc vc1</pre>

## mvmctl

### Synopsis

[mvmctl -d](#) [--flags=force] <vmid>

[mvmctl -f](#) <migrationPolicy> [--flags=eval [--xml]]

[mvmctl -m](#) [<options>] <vmid>

[mvmctl -M](#) dsthost=<newhost><vmid>

[mvmctl -q](#) <vmid> [--blocking] [--xml]

[mvmctl -w](#) state=drained

### Overview

mvmctl controls the modification, querying, migration, and destruction of virtual machines (VMs).

## Format

-d	
<b>Name</b>	Destroy
<b>Format</b>	<code>mvmctl -d [--flags=force] &lt;vmid&gt;</code>
<b>Description</b>	Destroys the specified VM. When you add the force flag, Moab forces the deletion of the VM if and only if it does not have a VM-tracking job.
<b>Example</b>	<pre>&gt; mvmctl -d oldVM</pre> <pre>&gt; mvmctl -d --flags=force oldVM</pre> <p><i>Because oldVM does not have a VM-tracking job associated with it and you set the force flag, Moab forces the deletion of oldVM.</i></p>

-f	
<b>Name</b>	Force Migrate
<b>Format</b>	<code>mvmctl -f consolidation overcommit [--flags=eval [--xml]]</code>
<b>Description</b>	Forces the migration policy on the system. The <code>eval</code> flag causes Moab to run through migration routines and report the results without actually migrating the VMs.
<b>Example</b>	<pre>&gt; mvmctl -f consolidation --flags=eval</pre> <p>Moab returns a report like the following:</p> <pre>1: VM 'vm1' from 'h0' to 'h3' 2: VM 'vm2' from 'h0' to 'h5'</pre>

-m	
<b>Name</b>	Modify
<b>Format</b>	<p><code>[&lt;options&gt;] &lt;vmid&gt;</code></p> <p>The <code>&lt;options&gt;</code> variable is a comma-separated list of <code>&lt;attr&gt;=&lt;value&gt;</code> pairs.</p>

-m	
<b>Description</b>	Modifies the VM.
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m gevent=hitemp:'mymessage' myNewVM</pre> <p><i>Gevents can be set using gevent.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m gmetric=bob:5.6 myNewVM</pre> <p><i>Gmetrics can be set using gmetric.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m os=compute myNewVM</pre> <p><i>Reprovisioning is done by changing os.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m powerstate=off myNewVM</pre> <p><i>Power management is done by modifying powerstate.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m variable=user:bob+purpose:myVM myNewVM</pre> <p><i>The modify variable uses the same syntax as Create.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>&gt; mvmctl -m flags=cannotmigrate myNewVM</pre> <p><i>Allow a VM to migrate by setting the canmigrate flag.</i></p> </div> <div style="border: 1px dashed gray; padding: 5px;"> <pre>&gt; mvmctl -m flags=canmigrate myNewVM</pre> <p><i>Allows a VM to migrate by setting the canmigrate flag.</i></p> </div>
<b>Notes</b>	<ul style="list-style-type: none"> <li>The variable option is a set-only operation. Previous variables will be overwritten.</li> </ul>

-M	
<b>Name</b>	Migrate
<b>Format</b>	dsthost=<newhost><vmid>

-M	
<b>Description</b>	<p>Migrate the given VM to the destination host.</p> <p>When you set the <i>vmid</i> to ANY, Moab migrates the VM to any available eligible hypervisor. For this to work, the following conditions must be met:</p> <ul style="list-style-type: none"> <li>• The VM reports a <b>CPULOAD</b>, and it is greater than 0.</li> <li>• The VM's <b>AMEMORY</b> is less than its <b>CMEMORY</b>. This indicates that some memory is currently in use and tells Moab that the RM is reporting memory correctly.</li> <li>• The VM's state is not "Unknown."</li> <li>• All hypervisors report a <b>CPULOAD</b>, and it is greater than 0.</li> <li>• All hypervisors report an <b>AMEMORY</b>, and it is less than its <b>CMEMORY</b>.</li> <li>• All hypervisors report a hypervisor type.</li> </ul>
<b>Example</b>	<pre>&gt; mvmctl -M dsthost=node05 myNewVM</pre> <p><i>myNewVM migrates to node05.</i></p> <pre>&gt; mvmctl -M dsthost=ANY vm42</pre> <p><i>Moab migrates vm42 to a node based on policy destination limitations (such as the <a href="#">NoVMMigrations</a> flag).</i></p>

-q	
<b>Name</b>	Query
<b>Format</b>	<vmid> [--blocking] [--xml]
<b>Description</b>	<p>Queries the specified VM; that is, it returns detailed information about the given VM. May be used with or without the <code>--xml</code> flag. ALL may also be used to display information about all VMs. This option gathers information from the Moab cache which prevents it from waiting for the scheduler, but the <code>--blocking</code> option can be used to bypass the cache and allow waiting for the scheduler.</p>
<b>Example</b>	<pre>&gt; mvmctl -q myNewVM</pre> <pre>&gt; mvmctl -q ALL --blocking</pre> <pre>&gt; mvmctl -q ALL --xml</pre>

-w	
<b>Name</b>	Constraint
<b>Format</b>	state=drained
<b>Description</b>	Overrides the HIDE DRAINED <a href="#">DISPLAYFLAGS</a> attribute allowing display of VMs in a <a href="#">DRAINED</a> state.
<b>Example</b>	<pre>&gt; mvmctl -q -w state=drained</pre>

## showbf

### Synopsis

`showbf [-A] [-a account] [-c class] [-d duration] [-D] [-f features] [-g group] [-L] [-m [=|>|>=|<|<=] memory] [-n nodecount] [-p partition] [-q qos] [-u user] [-v] [--blocking]`

### Overview

Shows what resources are available for immediate use.

**i** The results Moab returns do not include resources that may be freed due to preemption.

This command can be used by any user to find out how many processors are available for immediate use on the system. It is anticipated that users will use this information to submit jobs that meet these criteria and thus obtain quick job turnaround times. This command incorporates down time, reservations, and node state information in determining the available backfill window.

**i** If specific information is not specified, `showbf` will return information for the user and group running but with global access for other credentials. For example, if `-q qos` is not specified, Moab will return resource availability information for a job as if it were entitled to access all QOS based resources (i.e., resources covered by reservations with a QOS based ACL), if `-c class` is not specified, the command will return information for resources accessible by any class.

**i** The `showbf` command incorporates node configuration, node utilization, node state, and node reservation information into the results it reports. This command does not incorporate constraints imposed by credential based fairness policies on the results it reports.

### Access

By default, this command can be used by any user or administrator.

## Parameters

Parameter	Description
<b>ACCOUNT</b>	Account name.
<b>CLASS</b>	Class/queue required.
<b>DURATION</b>	Time duration specified as the number of seconds or in [DD:]HH:MM:SS notation.
<b>FEATURELIST</b>	Colon separated list of node features required.
<b>GROUP</b>	Specify particular group.
<b>MEMCMP</b>	Memory comparison used with the <code>-m</code> flag. Valid signs are <code>&gt;</code> , <code>&gt;=</code> , <code>==</code> , <code>&lt;=</code> , and <code>&lt;</code> .
<b>MEMORY</b>	Specifies the amount of required real memory configured on the node, (in MB), used with the <code>-m</code> flag.
<b>NODECOUNT</b>	Specify number of nodes for inquiry with <code>-n</code> flag.
<b>PARTITION</b>	Specify partition to check with <code>-p</code> flag.
<b>QOS</b>	Specify QOS to check with <code>-q</code> flag.
<b>USER</b>	Specify particular user to check with <code>-u</code> flag.

## Flags

Flag	Description
<b>-A</b>	Show resource availability information for all users, groups, and accounts. By default, <code>showbf</code> uses the default user, group, and account ID of the user issuing the command.
<b>-a</b>	Show resource availability information only for specified account.
<b>--blocking</b>	Do not use cache information in the output. The <code>--blocking</code> flag retrieves results exclusively from the scheduler.
<b>-d</b>	Show resource availability information for specified duration.

Flag	Description
<b>-D</b>	Display current and future resource availability notation.
<b>-g</b>	Show resource availability information only for specified group.
<b>-h</b>	Help for this command.
<b>-L</b>	Enforce Hard limits when showing available resources.
<b>-m</b>	Allows user to specify the memory requirements for the backfill nodes of interest. It is important to note that if the optional MEMCMP and MEMORY parameters are used, they must be enclosed in single ticks (') to avoid interpretation by the shell. For example, enter <code>showbf -m '==256'</code> to request nodes with 256 MB memory.
<b>-n</b>	Show resource availability information for a specified number of nodes. That is, this flag can be used to force <code>showbf</code> to display only blocks of resources with at least this many nodes available.
<b>-p</b>	Show resource availability information for the specified partition.
<b>-q</b>	Show information for the specified QOS.
<b>-r</b>	Show resource availability for the specified processor count.
<b>-u</b>	Show resource availability information only for specified user.

## Examples

### Example 3-51:

In this example, a job requiring up to 2 processors could be submitted for immediate execution in partition `ClusterA` for any duration. Additionally, a job requiring 1 processor could be submitted for immediate execution in partition `ClusterB`. Note that by default, each task is tracked and reported as a request for a single processor.

```
> showbf
Partition      Tasks  Nodes  StartOffset  Duration  StartDate
-----
ALL            3      3      00:00:00     INFINITY  11:32:38_08/19
ReqID=0
ClusterA      1      1      00:00:00     INFINITY  11:32:38_08/19
ReqID=0
ClusterB      2      2      00:00:00     INFINITY  11:32:38_08/19
ReqID=0
```

 StartOffset is the amount of time remaining before resources will be available.

**Example 3-52:**

In this example, the output verifies that a backfill window exists for jobs requiring a 3 hour runtime and at least 16 processors. Specifying job duration is of value when time based access is assigned to reservations (i.e., using the **SRCFG TIMELIMIT ACL**)

```
> showbf -r 16 -d 3:00:00
Partition    Tasks    Nodes    Duration    StartOffset    StartDate
-----
ALL          20       20       INFINITY    00:00:00       09:22:25_07/19
```

**Example 3-53:**

In this example, a resource availability window is requested for processors located only on nodes with at least 512 MB of memory.

```
> showbf -m '=512'
Partition    Tasks    Nodes    Duration    StartOffset    StartDate
-----
ALL          20       20       INFINITY    00:00:00       09:23:23_07/19
ClusterA     10       10       INFINITY    00:00:00       09:23:23_07/19
ClusterB     10       10       INFINITY    00:00:00       09:23:23_07/19
```

**Related topics**

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [showq](#)
- [mdiag -t](#)

## showq

### Synopsis

```
showq [-b] [-g] [-l] [-c|-i|-r] [-n] [-o] [-p partition] [-R rsvid] [-u] [-v] [-w <CONSTRAINT>] [--blocking] [--noblock]
```

### Overview

Displays information about active, eligible, blocked, and/or recently completed jobs. Since the resource manager is not actually scheduling jobs, the job ordering it displays is not valid. The `showq` command displays the actual job ordering under the Moab Workload Manager. When used without flags, this command displays all jobs in active, idle, and non-queued states.

### Access

By default, this command can be run by any user. However, the `-c`, `-i`, and `-r` flags can only be used by level 1, 2, or 3 Moab administrators.

## Flags

Flag	Description
<b>-b</b>	Display blocked jobs only
<b>-c</b>	Display details about recently <a href="#">completed</a> jobs (see <a href="#">example</a> , <a href="#">JOBPURGETIME</a> ).
<b>-g</b>	Display grid job and system IDs for all jobs.
<b>-i</b>	Display extended details about idle jobs. (see <a href="#">example</a> )
<b>-l</b>	Display local/remote view. For use in a <a href="#">Grid</a> environment, displays job usage of both local and remote compute resources.
<b>-n</b>	Displays normal showq output, but lists job names under JOBID
<b>-o</b>	Displays jobs in the active queue in the order specified (uses format <code>showq -o &lt;specifiedOrder&gt;</code> ). Valid options include REMAINING, REVERSEORDER, JOB, USER, STATE, and STARTTIME. The default is REMAINING.
<b>-p</b>	Display only jobs assigned to the specified partition.
<b>-r</b>	Display extended details about active (running) jobs. (see <a href="#">example</a> )
<b>-R</b>	Display only jobs which overlap the specified reservation.
<b>-u</b>	Display all running jobs for a particular user.
<b>-v</b>	Display local and full resource manager job IDs as well as partitions. If specified with the <code>-i</code> option, will display job reservation time. The <code>-v</code> option displays all array subjobs. All <code>showq</code> commands without the <code>-v</code> option show just the master jobs in an array.
<b>-w</b>	Display only jobs associated with the specified constraint. Valid constraints include user, group, acct, class, and qos (see <a href="#">showq -w</a> example.).
<b>--blocking</b>	Do not use cache information in the output. The <code>--blocking</code> flag retrieves results exclusively from the scheduler.
<b>--noblock</b>	Use cache information for a faster response.

## Details

Beyond job information, the `showq` command will also report if the scheduler is stopped or paused or if a system reservation is in place. Further, the `showq` command will also report public system messages.

## Examples

- [Default Report on page 323](#)
  - [Detailed Active/Running Job Report on page 326](#)
  - [Eligible Jobs on page 325](#)
  - [Detailed Completed Job Report on page 329](#)
- [Filtered Job Report on page 330](#)

*Example 3-54: Default Report*

The output of this command is divided into three parts, [Active](#) Jobs, [Eligible](#) Jobs, and [Blocked](#) Jobs.

```

> showq

active jobs-----
JOBID USERNAME STATE PROCS REMAINING          STARTTIME
12941      sartois   Running   25    2:44:11  Thu Sep  1 15:02:50
12954      t gates   Running    4    2:57:33  Thu Sep  1 15:02:52
12944      eval1    Running   16    6:37:31  Thu Sep  1 15:02:50
12946      t gates   Running    2    1:05:57:31 Thu Sep  1 15:02:50

4 active jobs          47 of 48 processors active (97.92%)
                      32 of 32 nodes active      (100.00%)

eligible jobs-----
JOBID      USERNAME      STATE  PROCS    WCLIMIT          QUEUE TIME
12956      cfosdyke     Idle   32    6:40:00  Thu Sep  1 15:02:50
12969      cfosdyke     Idle    4    6:40:00  Thu Sep  1 15:03:23
12939      eval1       Idle   16    3:00:00  Thu Sep  1 15:02:50
12940      mwillis     Idle    2    3:00:00  Thu Sep  1 15:02:50
12947      mwillis     Idle    2    3:00:00  Thu Sep  1 15:02:50
12949      eval1       Idle    2    3:00:00  Thu Sep  1 15:02:50
12953      t gates     Idle   10    4:26:40  Thu Sep  1 15:02:50
12955      eval1       Idle    2    4:26:40  Thu Sep  1 15:02:50
12957      t gates     Idle   16    3:00:00  Thu Sep  1 15:02:50
12963      eval1       Idle   16    1:06:00:00 Thu Sep  1 15:02:52
12964      t gates     Idle   16    1:00:00:00 Thu Sep  1 15:02:52
12937      allendr     Idle    9    1:00:00:00 Thu Sep  1 15:02:50
12962      aacker      Idle    6    00:26:40  Thu Sep  1 15:02:50
12968      tamaker     Idle    1    4:26:40  Thu Sep  1 15:02:52

14 eligible jobs

blocked jobs-----
JOBID      USERNAME      STATE  PROCS    WCLIMIT          QUEUE TIME

0 blocked jobs

Total jobs: 18

```

The fields are as follows:

Column	Description
<b>JOBID</b>	Job identifier.
<b>USERNAME</b>	<a href="#">User</a> owning job.
<b>STATE</b>	<a href="#">Job State</a> . Current batch state of the job.
<b>PROCS</b>	Number of processors being used by the job.

Column	Description
<b>REMAINING/WCLIMIT</b>	For active jobs, the time the job has until it has reached its wallclock limit or for idle/blocked jobs, the amount of time requested by the job. Time specified in [DD:]HH:MM:SS notation.
<b>STARTTIME</b>	Time job started running.

### Active Jobs

Active jobs are those that are [Running](#) or [Starting](#) and consuming resources. Displayed are the job id\*, the job's owner, and the job state. Also displayed are the number of processors allocated to the job, the amount of time remaining until the job completes (given in HH:MM:SS notation), and the time the job started. All active jobs are sorted in "Earliest Completion Time First" order.

**i** \*Job IDs may be marked with a single character to specify the following conditions:

Character	Description
<b>_ (underbar)</b>	job violates usage limit
<b>* (asterisk)</b>	job is backfilled AND is preemptible
<b>+ (plus)</b>	job is backfilled AND is NOT preemptible
<b>- (hyphen)</b>	job is NOT backfilled AND is preemptible

**i** Detailed active job information can be obtained using the `-r` flag.

### Eligible Jobs

Eligible Jobs are those that are queued and eligible to be scheduled. They are all in the Idle job state and do not violate any fairness policies or have any job holds in place. The jobs in the Idle section display the same information as the Active Jobs section except that the wallclock CPULIMIT is specified rather than job time REMAINING, and job QUEUE TIME is displayed rather than job STARTTIME. The jobs in this section are ordered by job priority. Jobs in this queue are considered eligible for both scheduling and backfilling.

**i** Detailed eligible job information can be obtained using the `-i` flag.

### Blocked Jobs

Blocked jobs are those that are ineligible to be run or queued. Jobs listed here could be in a number of states for the following reasons:

State	Description
<b>Idle</b>	Job violates a fairness policy. Use <code>diagnose -q</code> for more information.
<b>UserHold</b>	A user hold is in place.
<b>SystemHold</b>	An administrative or system hold is in place.
<b>BatchHold</b>	A scheduler batch hold is in place (used when the job cannot be run because the requested resources are not available in the system or because the resource manager has repeatedly failed in attempts to start the job).
<b>Deferred</b>	A scheduler defer hold is in place (a temporary hold used when a job has been unable to start after a specified number of attempts. This hold is automatically removed after a short period of time).
<b>NotQueued</b>	Job is in the resource manager state NQ (indicating the job's controlling scheduling daemon is unavailable).

A summary of the job queue's status is provided at the end of the output.

*Example 3-55: Detailed Active/Running Job Report*

```

> showq -r
active jobs-----
JOBID          S  PAR  EFFIC  XFACTOR  Q      USER  GROUP      MHOST  PROCS
REMAINING          STARTTIME
12941          R    3 100.00    1.0 -   sartois  Arches      G5-014  25
2:43:31 Thu Sep 1 15:02:50
12954          R    3 100.00    1.0 Hi   t gates  Arches      G5-016   4
2:56:54 Thu Sep 1 15:02:52
12944          R    2 100.00    1.0 De   e val1  RedRock    P690-016 16
6:36:51 Thu Sep 1 15:02:50
12946          R    3 100.00    1.0 -   t gates  Arches      G5-001   2
1:05:56:51 Thu Sep 1 15:02:50

4 active jobs          47 of 48 processors active (97.92%)
                      32 of 32 nodes active      (100.00%)

Total jobs: 4
    
```

The fields are as follows:

Column	Description
<b>JOBID</b>	Name of active job.

Column	Description
<b>S</b>	<a href="#">Job State</a> . Either R for Running or S for Starting.
<b>PAR</b>	Partition in which job is running.
<b>EFFIC</b>	CPU efficiency of job.
<b>XFACTOR</b>	Current expansion factor of job, where $XFactor = (QueueTime + WallClockLimit) / WallClockLimit$
<b>Q</b>	Quality Of Service specified for job.
<b>USERNAME</b>	User owning job.
<b>GROUP</b>	Primary group of job owner.
<b>MHOST</b>	Master Host running primary task of job.
<b>PROCS</b>	Number of processors being used by the job.
<b>REMAINING</b>	Time the job has until it has reached its wallclock limit. Time specified in HH:MM:SS notation.
<b>STARTTIME</b>	Time job started running.

After displaying the running jobs, a summary is provided indicating the number of jobs, the number of allocated processors, and the system utilization.

Column	Description
<b>JobName</b>	Name of active job.
<b>S</b>	Job State. Either R for Running or S for Starting.
<b>CCode</b>	Completion Code. The return/completion code given when a job completes. (Only applicable to completed jobs.)
<b>Par</b>	Partition in which job is running.
<b>Effic</b>	CPU efficiency of job.
<b>XFactor</b>	Current expansion factor of job, where $XFactor = (QueueTime + WallClockLimit) / WallClockLimit$

Column	Description
<b>Q</b>	Quality Of Service specified for job.
<b>User</b>	User owning job.
<b>Group</b>	Primary group of job owner.
<b>Nodes</b>	Number of processors being used by the job.
<b>Remaining</b>	Time the job has until it has reached its wallclock limit. Time specified in HH:MM:SS notation.
<b>StartTime</b>	Time job started running.

```

> showq -i
eligible jobs-----
JOBID          PRIORITY  XFACTOR  Q      USER   GROUP  PROCS  WCLIMIT
CLASS          SYSTEMQUEUE TIME
12956*
batch Thu Sep 1 15:02:50 20      1.0 - cfosdyke RedRock 32      6:40:00
12969*
batch Thu Sep 1 15:03:23 19      1.0 - cfosdyke RedRock 4        6:40:00
12939
batch Thu Sep 1 15:02:50 16      1.0 - evall   RedRock 16      3:00:00
12940
batch Thu Sep 1 15:02:50 16      1.0 - mwillis Arches  2        3:00:00
12947
batch Thu Sep 1 15:02:50 16      1.0 - mwillis Arches  2        3:00:00
12949
batch Thu Sep 1 15:02:50 16      1.0 - evall   RedRock 2        3:00:00
12953
batch Thu Sep 1 15:02:50 16      1.0 - tgates  Arches 10      4:26:40
12955
batch Thu Sep 1 15:02:50 16      1.0 - evall   RedRock 2        4:26:40
12957
batch Thu Sep 1 15:02:50 16      1.0 - tgates  Arches 16      3:00:00
12963
batch Thu Sep 1 15:02:52 16      1.0 - evall   RedRock 16      1:06:00:00
12964
batch Thu Sep 1 15:02:52 16      1.0 - tgates  Arches 16      1:00:00:00
12937
batch Thu Sep 1 15:02:50 1        1.0 - allendr RedRock 9        1:00:00:00
12962
batch Thu Sep 1 15:02:50 1        1.2 - aacker  RedRock 6        00:26:40
12968
batch Thu Sep 1 15:02:52 1        1.0 - tamaker RedRock 1        4:26:40

14 eligible jobs
Total jobs: 14
    
```

The fields are as follows:

Column	Description
<b>JOBID</b>	Name of job.
<b>PRIORITY</b>	Calculated job priority.
<b>XFACTOR</b>	Current expansion factor of job, where XFactor = (QueueTime + WallClockLimit) / WallClockLimit
<b>Q</b>	Quality Of Service specified for job.
<b>USER</b>	<a href="#">User</a> owning job.
<b>GROUP</b>	Primary <a href="#">group</a> of job owner.
<b>PROCS</b>	Minimum number of processors required to run job.
<b>WCLIMIT</b>	Wallclock limit specified for job. Time specified in HH:MM:SS notation.
<b>CLASS</b>	<a href="#">Class</a> requested by job.
<b>SYSTEMQUEUE TIME</b>	Time job was admitted into the system queue.

**i** An asterisk at the end of a job (job 12956\* in this example) indicates that the job has a [job reservation](#) created for it. The details of this reservation can be displayed using the [checkjob](#) command.

**Example 3-56: Detailed Completed Job Report**

```

> showq -c
completed jobs-----
JOBID      PROC      WALLTIME  SCCODE  PAR  EFFIC  XFACTOR  Q  USERNAME  GROUP  MHOST
13098     25      2:43:31  Thu Sep 1 15:02:50  C    0  bas  93.17    1.0  -   sartois  Arches  G5-014
13102     4       2:56:54  Thu Sep 1 15:02:52  C    0  bas  99.55    2.2  Hi   tgates   Arches  G5-016
13103    16       6:36:51  Thu Sep 1 15:02:50  C    2  tes  99.30    2.9  De   evall    RedRock P690-016
13115     2       1:05:56:51 Thu Sep 1 15:02:50  C    0  tes  97.04    1.0  -   tgates   Arches  G5-001
3 completed jobs
    
```

The fields are as follows:

Column	Description
<b>JOBID</b>	job id for completed job.
<b>S</b>	<a href="#">Job State</a> . Either C for <a href="#">Completed</a> or V for <a href="#">Vacated</a> .
<b>CCODE</b>	Completion code reported by the job.
<b>PAR</b>	Partition in which job ran.
<b>EFFIC</b>	CPU efficiency of job.
<b>XFACTOR</b>	Expansion factor of job, where $XFactor = (QueueTime + WallClockLimit) / WallClockLimit$
<b>Q</b>	<a href="#">Quality of Service</a> specified for job.
<b>USERNAME</b>	<a href="#">User</a> owning job.
<b>GROUP</b>	Primary <a href="#">group</a> of job owner.
<b>MHOST</b>	Master Host which ran the primary task of job.
<b>PROCS</b>	Number of processors being used by the job.
<b>WALLTIME</b>	Wallclock time used by the job. Time specified in [DD:]HH:MM:SS notation.
<b>STARTTIME</b>	Time job started running.

After displaying the active jobs, a summary is provided indicating the number of jobs, the number of allocated processors, and the system utilization.

**i** If the [DISPLAYFLAGS](#) parameter is set to *ACCOUNTCENTRIC*, job group information will be replaced with job account information.

#### Example 3-57: Filtered Job Report

Show only jobs associated with user john and class benchmark.

```
> showq -w class=benchmark -w user=john
...
```

## Job Array

Job arrays show the name of the job array and then in parenthesis, the number of sub-jobs in the job array that are in the specified state.

```
> showq

active jobs-----
JOBID            USERNAME        STATE  PROCS   REMAINING        STARTTIME
Moab.1(14)       aesplin         Running  14     00:59:41  Fri May 27 14:58:57

14 active jobs           14 of 14 processors in use by local jobs (100.00%)
2 of 2 nodes active     (100.00%)

eligible jobs-----
JOBID            USERNAME        STATE  PROCS   WCLIMIT        QUEUETIME
Moab.1(4)        aesplin         Idle    4       1:00:00  Fri May 27 14:58:52

4 eligible jobs

blocked jobs-----
JOBID            USERNAME        STATE  PROCS   WCLIMIT        QUEUETIME
Moab.1(2)        aesplin         Blocked  2       1:00:00  Fri May 27 14:58:52

2 blocked jobs

Total jobs: 20
```

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [showbf](#) - command to display resource availability.
- [mdiag -j](#) - command to display detailed job diagnostics.
- [checkjob](#) - command to check the status of a particular job.
- [JOBBCPURGETIME](#) - parameter to adjust the duration of time Moab preserves information about completed jobs
- [DISPLAYFLAGS](#) - parameter to control what job information is displayed

## showhist.moab.pl

### Synopsis

```
showhist.moab.pl [-a accountname]
                 [-c classname] [-e enddate]
                 [-g groupname] [-j jobid] [-n days]
                 [-q qosname] [-s startdate]
                 [-u username]
```

## Overview

The `showhist.moab.pl` script displays historical job information. Its purpose is similar to the [checkjob](#) command's, but `showhist.moab.pl` displays information about jobs that have already completed.

## Access

By default, this script's use is limited to administrators on the head node; however, end users can also be given power to run the script. To grant access to the script to end users, move `showhist.moab.pl` from the `tools` directory to the `bin` directory.

## Arguments

<b>-a (Account)</b>	
<b>Format</b>	<ACCOUNTNAME>
<b>Description</b>	Displays job records matching the specified account.
<b>Example</b>	<pre>&gt; showhist.moab.pl -a myAccount</pre> <p><i>Information about jobs related to the account myAccount is displayed.</i></p>

<b>-c (Class)</b>	
<b>Format</b>	<CLASSNAME>
<b>Description</b>	Displays job records matching the specified class (queue).
<b>Example</b>	<pre>&gt; showhist.moab.pl -c newClass</pre> <p><i>Information about jobs related to the class newClass is displayed.</i></p>

<b>-e (End Date)</b>	
<b>Format</b>	YYYY-MM-DD
<b>Description</b>	Displays the records of jobs recorded before or on the specified date.

-e (End Date)	
<b>Example</b>	<pre>&gt; showhist.moab.pl -e 2001-01-03</pre> <p><i>Information about all jobs recorded on or before January 3, 2001 is displayed.</i></p> <pre>&gt; showhist.moab.pl -s 2011-01-01 -e 2011-01-31</pre> <p><i>Information is displayed about all jobs recorded in January 2011.</i></p>

-g (Group)	
<b>Format</b>	<GROUPNAME>
<b>Description</b>	Displays job records matching the specified group.
<b>Example</b>	<pre>&gt; showhist.moab.pl -g admins</pre> <p><i>Information about jobs related to the group admins is displayed.</i></p>

-j (Job ID)	
<b>Format</b>	<JOBID>
<b>Description</b>	Displays job records matching the specified job id.
<b>Example</b>	<pre>&gt; showhist.moab.pl -j moab01</pre> <p><i>Information about job moab01 is displayed.</i></p>

-n (Number of Days)	
<b>Format</b>	<INTEGER>

-n (Number of Days)	
<b>Description</b>	Restricts the number of past jobs to search by a specified number of days relative to today.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; showhist.moab.pl -n 90 -j moab924</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Displays job information for job moab924. The search is restricted to the last 90 days.</i></p>

-q (QoS)	
<b>Format</b>	<QOSNAME>
<b>Description</b>	Displays job records matching the specified quality of service.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; showhist.moab.pl -q myQoS</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Information about jobs related to the QoS myQoS is displayed.</i></p>

-s (Start Date)	
<b>Format</b>	YYYY-MM-DD
<b>Description</b>	Displays the records of jobs that recorded on the specified date and later.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; showhist.moab.pl -s 1776-07-04</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Information about all jobs recorded on July 4, 1776 and later is displayed.</i></p> <pre style="border: 1px dashed gray; padding: 5px; margin-top: 10px;">&gt; showhist.moab.pl -s 2001-07-05 -e 2002-07-05</pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 5px;"><i>Information is displayed about all jobs recorded between July 5, 2001 and July 5, 2002.</i></p>

<b>-u (User)</b>	
<b>Format</b>	<USERNAME>
<b>Description</b>	Displays job records matching the specified user.
<b>Example</b>	<pre>&gt; showhist.moab.pl -u bob</pre> <p><i>Information about user bob's jobs is displayed.</i></p>

### Sample Output

```
> showhist.moab.pl
Job Id           : Moab.4
User Name        : user1
Group Name       : company
Queue Name       : NONE
Processor Count  : 4
Wallclock Duration: 00:00:00
Submit Time      : Mon Nov 21 10:48:32 2011
Start Time       : Mon Nov 21 10:49:37 2011
End Time         : Mon Nov 21 10:49:37 2011
Exit Code        : 0
Allocated Nodelist: 10.10.10.3

Job Id           : Moab.1
Executable       : 4
User Name        : user1
Group Name       : company
Account Name     : 1321897709
Queue Name       : NONE
Quality Of Service: 0M
Processor Count  : -0
Wallclock Duration: 00:01:05
Submit Time      : Mon Nov 21 10:48:29 2011
Start Time       : Mon Nov 21 10:48:32 2011
End Time         : Mon Nov 21 10:49:37 2011
Exit Code        : 0
Allocated Nodelist: 512M
```

Information is displayed for all completed jobs.



When a job's Start Time and End Time are the same, the job is infinite and still running.

### Related topics

- [checkjob](#) - explains how to query for a status report for a specified job.
- [mdiag -j](#) command - display additional detailed information regarding jobs
- [showq](#) command - showq high-level job summaries

## showres

### Synopsis

```
showres [-f] [-n [-g]] [-o] [-r] [reservationid]
```

### Overview

This command displays all reservations currently in place within Moab. The default behavior is to display reservations on a reservation-by-reservation basis.

### Access

By default, this command can be run by any Moab administrator.

Flag	Description
<b>-f</b>	Show free (unreserved) resources rather than reserved resources. The <code>-f</code> flag cannot be used in conjunction with the any other flag
<b>-g</b>	When used with the <code>-n</code> flag, shows <code>grep</code> -able output with nodename on every line
<b>-n</b>	Display information regarding all nodes reserved by <code>&lt;RSVID&gt;</code>
<b>-o</b>	Display all reservations which overlap <code>&lt;RSVID&gt;</code> (in time and space) <div style="border: 1px solid #ccc; border-radius: 5px; padding: 5px; margin-top: 5px;">  Not supported with <code>-n</code> flag         </div>
<b>-r</b>	Display reservation timeframes in relative time mode
<b>-v</b>	Show verbose output. If used with the <code>-n</code> flag, the command will display all reservations found on nodes contained in <code>&lt;RSVID&gt;</code> . Otherwise, it will show long reservation start dates including the reservation year.

Parameter	Description
<b>RSVID</b>	ID of reservation of interest — optional

## Examples

Example 3-58:

```

> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
12941              Job R      -00:05:01  2:41:39  2:46:40  13/25    Thu Sep 1
15:02:50
12944              Job R      -00:05:01  6:34:59  6:40:00  16/16    Thu Sep 1
15:02:50
12946              Job R      -00:05:01  1:05:54:59  1:06:00:00  1/2      Thu Sep 1
15:02:50
12954              Job R      -00:04:59  2:55:01  3:00:00  2/4      Thu Sep 1
15:02:52
12956              Job I      1:05:54:59  1:12:34:59  6:40:00  16/32    Fri Sep 2
21:02:50
12969              Job I      6:34:59    13:14:59  6:40:00  4/4      Thu Sep 1
21:42:50

6 reservations located
    
```

*The above example shows all reservations on the system.*

The fields are as follows:

Column	Description
<b>Type</b>	Reservation Type. This will be one of the following: Job or User.
<b>ReservationID</b>	This is the name of the reservation. Job reservation names are identical to the job name. User, Group, or Account reservations are the user, group, or account name followed by a number. System reservations are given the name SYSTEM followed by a number.
<b>S</b>	State. This field is valid only for job reservations. It indicates whether the job is (S)tarting, (R)unning, or (I)dle.
<b>Start</b>	Relative start time of the reservation. Time is displayed in HH:MM:SS notation and is relative to the present time.
<b>End</b>	Relative end time of the reservation. Time is displayed in HH:MM:SS notation and is relative to the present time. Reservations that will not complete in 1,000 hours are marked with the keyword INFINITY.
<b>Duration</b>	Duration of the reservation in HH:MM:SS notation. Reservations lasting more than 1,000 hours are marked with the keyword INFINITY.
<b>Nodes</b>	Number of nodes involved in reservation.

Column	Description
<b>StartTime</b>	Time Reservation became active.

*Example 3-59:*

```

> showres -n
reservations on Thu Sep 1 16:49:59

NodeName      Type      ReservationID  JobState Task      Start      Duration
StartTime

G5-001        Job      12946         Running  2      -1:47:09  1:06:00:00  Thu
Sep 1 15:02:50
G5-001        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-002        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-002        Job      12953         Running 2      -00:29:37  4:26:40     Thu
Sep 1 16:20:22
G5-003        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-003        Job      12953         Running 2      -00:29:37  4:26:40     Thu
Sep 1 16:20:22
G5-004        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-004        Job      12953         Running 2      -00:29:37  4:26:40     Thu
Sep 1 16:20:22
G5-005        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-005        Job      12953         Running 2      -00:29:37  4:26:40     Thu
Sep 1 16:20:22
G5-006        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-006        Job      12953         Running 2      -00:29:37  4:26:40     Thu
Sep 1 16:20:22
G5-007        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-007        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-008        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-008        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-009        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-009        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-010        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-010        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-011        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-011        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-012        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-012        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-013        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-013        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22
G5-014        Job      12956         Idle    2      1:04:12:51  6:40:00     Fri
Sep 2 21:02:50
G5-014        Job      12939         Running 2      -00:29:37  3:00:00     Thu
Sep 1 16:20:22

```

G5-015	Job	12956	Idle	2	1:04:12:51	6:40:00	Fri
Sep 2 21:02:50							
G5-015	Job	12949	Running	2	-00:08:57	3:00:00	Thu
Sep 1 16:41:02							
G5-016	Job	12956	Idle	2	1:04:12:51	6:40:00	Fri
Sep 2 21:02:50							
G5-016	Job	12947	Running	2	-00:08:57	3:00:00	Thu
Sep 1 16:41:02							
P690-001	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-002	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-003	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-004	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-005	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-006	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-007	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-008	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-009	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-010	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-011	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-012	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-013	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-013	Job	12969	Idle	1	4:52:51	6:40:00	Thu
Sep 1 21:42:50							
P690-014	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-014	Job	12969	Idle	1	4:52:51	6:40:00	Thu
Sep 1 21:42:50							
P690-015	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-015	Job	12969	Idle	1	4:52:51	6:40:00	Thu
Sep 1 21:42:50							
P690-016	Job	12944	Running	1	-1:47:09	6:40:00	Thu
Sep 1 15:02:50							
P690-016	Job	12969	Idle	1	4:52:51	6:40:00	Thu
Sep 1 21:42:50							
52 nodes reserved							

*This example shows reservations for nodes.*

The fields are as follows:

Column	Description
<b>NodeName</b>	Node on which reservation is placed.

Column	Description
<b>Type</b>	Reservation Type. This will be one of the following: Job or User.
<b>ReservationID</b>	This is the name of the reservation. Job reservation names are identical to the job name. User, Group, or Account reservations are the user, group, or account name followed by a number. System reservations are given the name SYSTEM followed by a number.
<b>JobState</b>	This field is valid only for job reservations. It indicates the state of the job associated with the reservation.
<b>Start</b>	Relative start time of the reservation. Time is displayed in HH:MM:SS notation and is relative to the present time.
<b>Duration</b>	Duration of the reservation in HH:MM:SS notation. Reservations lasting more than 1000 hours are marked with the keyword INFINITY.
<b>StartTime</b>	Time Reservation became active.

Example 3-60:

```
> showres 12956
ReservationID      Type S      Start      End      Duration  N/P      StartTime
12956              Job I    1:04:09:32 1:10:49:32 6:40:00   16/32    Fri Sep 2
21:02:50
1 reservation located
```

*In this example, information for a specific reservation (job) is displayed.*

## Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mrsvctl -c](#) - create new reservations.
- [mrsvctl -r](#) - release existing reservations.
- [mdiag -r](#) - diagnose/view the state of existing reservations.
- [Reservation Overview](#) - description of reservations and their use.

## showstart

### Synopsis

```
showstart {jobid|proccount[@duration]|s3jobspec} [-e {all|hist|prio|rsv}] [-f] [-g [peer]] [-l qos=<QOS>] [--  
blocking] [--format=xml]
```

## Overview

This command displays the estimated start time of a job based a number of analysis types. This analysis may include information based on historical usage, earliest available reservable resources, and priority based backlog analysis. Each type of analysis will provide somewhat different estimates based on current cluster environmental conditions. By default, only reservation based analysis is performed.

**i** The start time estimate Moab returns does not account for resources that will become available due to preemption.

**Historical** analysis utilizes historical queue times for jobs which match a similar processor count and job duration profile. This information is updated on a sliding window which is configurable within `moab.cfg`

**Reservation** based start time estimation incorporates information regarding current administrative, user, and job reservations to determine the earliest time the specified job could allocate the needed resources and start running. In essence, this estimate will indicate the earliest time the job would start assuming this job was the highest priority job in the queue.

**Priority** based job start analysis determines when the queried job would fit in the queue and determines the estimated amount of time required to complete the jobs which are currently running or scheduled to run before this job can start.

In all cases, if the job is running, this command will return the time the job started. If the job already has a reservation, this command will return the start time of the reservation.

## Access

By default, this command can be run by any user.

## Parameters

Parameter	Description
<b>--blocking</b>	Do not use cache information in the output. The <code>--blocking</code> flag retrieves results exclusively from the scheduler.
<b>DURATION</b>	Duration of pseudo-job to be checked in format <code>[[[DD:]HH:]MM:]SS</code> (default duration is 1 second)
<b>-e</b>	Estimate method. By default, Moab will use the reservation based estimation method.
<b>-f</b>	Use feedback. If specified, Moab will apply historical accuracy information to Improve the quality of the estimate. See <a href="#">ENABLESTARTESTIMATESTATS</a> for more information.

Parameter	Description
<b>-g</b>	<p>Grid mode. Obtain showstart information from remote resource managers. If -g is not used and Moab determines that job is already migrated, Moab obtains showstart information from the remote Moab where the job was migrated to. All resource managers can be queried by using the keyword "all" which returns all information in a table.</p> <pre style="border: 1px dashed gray; padding: 5px;">\$ showstart -g all head.1 Estimated Start Times [ Remote RM ] [ Reservation ] [ Priority ] [ Historical ] [ c1 ] [ 00:15:35 ] [ ] [ ] [ c2 ] [ 3:15:38 ] [ ] [ ]</pre>
<b>-l qos- s=&lt;QOS&gt;</b>	Specifies what QOS the job must start under, using the same syntax as the <a href="#">msub</a> command. Currently, no other resource manager extensions are supported. This flag only applies to hypothetical jobs by using the <code>proccount[@duration]</code> syntax.
<b>JOBID</b>	Job to be checked
<b>PROCCOUNT</b>	Number of processors in pseudo-job to be checked
<b>S3JOBSPEC</b>	XML describing the job according to the Dept. of Energy <a href="#">Scalable Systems Software/S3</a> job specification.

## Examples

### Example 3-61:

```
> showstart orion.13762
job orion.13762 requires 2 procs for 0:33:20
Estimated Rsv based start in          1:04:55 on Fri Jul 15 12:53:40
Estimated Rsv based completion in     2:44:55 on Fri Jul 15 14:33:40
Estimated Priority based start in      5:14:55 on Fri Jul 15 17:03:40
Estimated Priority based completion in  6:54:55 on Fri Jul 15 18:43:40
Estimated Historical based start in    00:00:00 on Fri Jul 15 11:48:45
Estimated Historical based completion in 1:40:00 on Fri Jul 15 13:28:45
Best Partition: fast
```

### Example 3-62:

```
> showstart 12@3600
job 12@3600 requires 12 procs for 1:00:00
Earliest start in          00:01:39 on Wed Aug 31 16:30:45
Earliest completion in    1:01:39 on Wed Aug 31 17:30:45
Best Partition: 32Bit
```

**i** You cannot specify job flags when running `showstart`, and since a job by default can only run on one partition, `showstart` fails when querying for a job requiring more nodes than the largest partition available.

## Additional Information

For reservation based estimates, the information provided by this command is more highly accurate if the job is highest priority, if the job has a reservation, or if the majority of the jobs which are of higher priority have reservations. Consequently, sites wishing to make decisions based on this information may want to consider using the [RESERVATIONDEPTH](#) parameter to increase the number of priority based reservations. This can be set so that most or even all idle jobs receive priority reservations and make the results of this command generally useful. The only caution of this approach is that increasing the **RESERVATIONDEPTH** parameter more tightly constrains the decisions of the scheduler and may resulting in slightly lower system utilization (typically less than 8% reduction).

### Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [checkjob](#)
- [showres](#)
- [showstats -f eststarttime](#)
- [showstats -f avgqtime](#)
- [Job Start Estimates](#)

## showstate

### Synopsis

`showstate`

### Overview

This command provides a summary of the state of the system. It displays a list of all active jobs and a text-based map of the status of all nodes and the jobs they are servicing. Basic diagnostic tests are also performed and any problems found are reported.

### Access

By default, this command can be run by any Moab Administrator.

## Examples

Example 3-63:

```
> showstate
cluster state summary for Wed Nov 23 12:00:21
  JobID          S      User      Group Procs    Remaining          StartTime
-----
(A)   fr17n11.942.0 R      johns   staff   16    13:21:15          Nov 22 12:00:21
(B)   fr17n12.942.0 S      johns   staff   32    13:07:11          Nov 22 12:00:21
(C)   fr17n13.942.0 R      johns   staff    8    11:22:25          Nov 22 12:00:21
(D)   fr17n14.942.0 S      johns   staff    8    10:43:43          Nov 22 12:01:21
(E)   fr17n15.942.0 S      johns   staff    8     9:19:25          Nov 22 12:01:21
(F)   fr17n16.942.0 R      johns   staff    8     9:01:16          Nov 22 12:01:21
(G)   fr17n17.942.0 R      johns   staff    1     7:28:25          Nov 22 12:03:22
(H)   fr17n18.942.0 R      johns   staff    1     3:05:17          Nov 22 12:04:22
(I)   fr17n19.942.0 S      johns   staff   24    0:54:38          Nov 22 12:00:22
Usage Summary:  9 Active Jobs 106 Active Nodes
                [0][0][0][0][0][0][0][0][0][1][1][1][1][1][1][1]
                [1][2][3][4][5][6][7][8][9][0][1][2][3][4][5][6]
Frame 2: XXXXXXXXXXXXXXXXXXXXXXXXXXXX[ ] [A] [C] [ ] [A] [C] [C] [A]
Frame 3: [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [A] [ ] [ ] [I] [ ] [I] [ ] [ ] [ ] [ ] [ ] [ ]
Frame 4: [ ] [I] [ ] [ ] [ ] [ ] [A] [ ] [ ] [I] [ ] [ ] [ ] [ ] [E] [ ] [I] [ ] [E]
Frame 5: [F] [ ] [E] [ ] [ ] [ ] [F] [F] [F] [I] [ ] [ ] [ ] [E] [ ] [E] [E]
Frame 6: [ ] [I] [I] [E] [I] [ ] [I] [I] [ ] [I] [F] [I] [I] [I] [I] [F]
Frame 7: [ ] XXX [ ] XXX [ ] XXX [ ] XXX [b] XXX [ ] XXX [ ] XXX [#] XXX
Frame 9: [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [E] [ ]
Frame 11: [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [I] [F] [0] [ ] [A] [I] [ ] [F] [ ] [A]
Frame 12: [A] [ ] [ ] [ ] [A] [ ] [ ] [ ] [C] [A] [ ] [ ] [C] [A] [A] [ ] [ ] [ ] [ ]
Frame 13: [D] XXX [I] XXX [ ] XXX [ ] XXX [ ] XXX [ ] XXX [I] XXX [I] XXX
Frame 14: [D] XXX [I] XXX [I] XXX [D] XXX [ ] XXX [H] XXX [I] XXX [ ] XXX
Frame 15: [b] XXX [b] XXX [b] XXX [b] XXX [D] XXX [b] XXX [b] XXX [b] XXX
Frame 16: [b] XXX [ ] XXX [b] XXX [ ] XXX [b] XXX [b] XXX [ ] XXX [b] XXX
Frame 17: [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ] [ ]
Frame 21: [ ] XXX [b] XXX [b] XXX [ ] XXX [b] XXX [b] XXX [b] XXX [b] XXX
Frame 22: [b] XXX [b] XXX [b] XXX [ ] XXX [b] XXX [ ] XXX [b] XXX [b] XXX
Frame 27: [b] XXX [b] XXX [ ] XXX [b] XXX [b] XXX [b] XXX [b] XXX [b] XXX
Frame 28: [G] XXX [ ] XXX [D] XXX [ ] XXX [D] XXX [D] XXX [D] XXX [ ] XXX
Frame 29: [A] [C] [A] [A] [C] [ ] [A] [C] XXXXXXXXXXXXXXXXXXXXXXXXXXXX
Key:  XXX:Unknown [*]:Down w/Job [#]:Down [']:Idle w/Job [ ]:Idle [@]:Busy w/No Job
      [!]:Drained
Key:  [a]:(Any lower case letter indicates an idle node that is assigned to a job)

Check Memory on Node fr3n07
Check Memory on Node fr4n06
Check Memory on Node fr4n09
```

*In this example, nine active jobs are running on the system. Each job listed in the top of the output is associated with a letter. For example, job fr17n11.942.0 is associated with the letter A. This letter can now be used to determine where the job is currently running. By looking at the system map, it can be found that job fr17n11.942.0 (job A) is running on nodes fr2n10, fr2n13, fr2n16, fr3n07...*

*The key at the bottom of the system map can be used to determine unusual node states. For example, fr7n15 is currently in the state down.*

*After the key, a series of warning messages may be displayed indicating possible system problems. In this case, warning messages indicate that there are memory problems on three nodes, fr3n07, fr4n06, and fr4n09. Also, warning messages indicate that job fr15n09.1097.0 is having difficulty starting. Node fr11n08 is in state BUSY but has no job assigned to it (it possibly has a runaway job running on it).*

## Related topics

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [Specifying Node Rack/Slot Location](#)

## showstats

### Synopsis

showstats

[showstats -a](#) [*accountid*] [-v] [-t <TIMESPEC>]

[showstats -c](#) [*classid*] [-v] [-t <TIMESPEC>]

[showstats -f](#) <*statistictype*>

[showstats -g](#) [*groupid*] [-v] [-t <TIMESPEC>]

[showstats -j](#) [*jobtemplate*] [-t <TIMESPEC>]

[showstats -n](#) [*nodeid*] [-t <TIMESPEC>]

[showstats -q](#) [*qosid*] [-v] [-t <TIMESPEC>]

[showstats -s](#)

[showstats -T](#) [*leafid* | *tree-level*]

[showstats -u](#) [*userid*] [-v] [-t <TIMESPEC>]

### Overview

This command shows various accounting and resource usage statistics for the system. Historical statistics cover the timeframe from the most recent execution of the [mschedctl -f](#) command.

### Access

By default, this command can be run by any Moab level 1, 2, or 3 Administrator.

### Parameters

Flag	Description
<b>-a</b> <ACCOUNTID>	Display account statistics. See <a href="#">Account statistics on page 349</a> for an example.
<b>-c</b> <CLASSID>	Display class statistics
<b>-f</b> <statistictype>	Display full matrix statistics (see <a href="#">showstats -f</a> for full details)

Flag	Description
<b>-g</b> [<GROUPID>]	Display group statistics. See <a href="#">Group statistics on page 351</a> for an example.
<b>-j</b> [<JOBTEMPLATE>]	Display template statistics
<b>-n</b> [<NODEID>]	Display node statistics ( <a href="#">ENABLEPROFILING</a> must be set). See <a href="#">Node statistics on page 353</a> for an example.
<b>-q</b> [<QOSID>]	Display QoS statistics
<b>-s</b>	display general scheduler statistics
<b>-t</b>	<p>Display statistical information from the specified timeframe:</p> <pre style="border: 1px dashed gray; padding: 5px;"> &lt;START_TIME&gt;[,&lt;END_TIME&gt;]   (ABSTIME: [HH[:MM[:SS]]][_MO[/DD[/YY]]] ie 14:30_06/20)   (RELTIME: -[[[DD:]HH:]MM:]SS) </pre> <p>See <a href="#">Statistics from an absolute time frame on page 359</a> and <a href="#">Statistics from a relative time frame on page 359</a> for examples.</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p><b>i</b> Profiling must be enabled for the credential type you want statistics for. See <a href="#">Credential Statistics</a> for information on how to enable profiling. Also, <b>-t</b> is not a stand-alone option. It must be used in conjunction with the <b>-a</b>, <b>-c</b>, <b>-g</b>, <b>-n</b>, <b>-q</b>, or <b>-u</b> flag.</p> </div>
<b>-T</b>	Display fairshare tree statistics. See <a href="#">Fairshare tree statistics on page 358</a> for an example.
<b>-u</b> [<USERID>]	Display user statistics. See <a href="#">User statistics on page 355</a> for an example.
<b>-v</b>	Display verbose information. See <a href="#">Verbose statistics on page 353</a> for an example.

## Examples

### Example 3-64: Account statistics

```
> showstats -a
Account Statistics Initialized Tue Aug 26 14:32:39
|----- Running -----|----- Completed -----
|-----|
Account      Jobs Procs ProcHours  Jobs   %   PHReq   %   PHDed   %   FSTgt  AvgXF
MaxXF  AvgQH  Effic  WCAcc
137651      16   92   1394.52  229  39.15  18486  45.26  7003.5  41.54  40.00  0.77
8.15   5.21  90.70  34.69
462212      11   63   855.27   43   7.35   6028  14.76  3448.4  20.45  6.25  0.71
5.40   3.14  98.64  40.83
462213       6   72   728.12   90  15.38   5974  14.63  3170.7  18.81  6.25  0.37
4.88   0.52  82.01  24.14
005810       3   24   220.72   77  13.16   2537   6.21  1526.6   9.06  -----  1.53
14.81   0.42  98.73  28.40
175436       0    0    0.00   12   2.05   6013  14.72   958.6   5.69  2.50  1.78
8.61   5.60  83.64  17.04
000102       0    0    0.00    1   0.17    64    0.16    5.1    0.03  -----  10.85
10.85  10.77  27.90  7.40
000023       0    0    0.00    1   0.17    12    0.03    0.2    0.00  -----  0.04
0.04   0.19  21.21  1.20
```

*This example shows a statistical listing of all active accounts. The top line (Account Statistics Initialized...) of the output indicates the beginning of the timeframe covered by the displayed statistics.*

*The statistical output is divided into two categories, Running and Completed. Running statistics include information about jobs that are currently running. Completed statistics are compiled using historical information from both running and completed jobs.*

The fields are as follows:

Column	Description
<b>Account</b>	Account Number
<b>Jobs</b>	Number of running jobs
<b>Procs</b>	Number of processors allocated to running jobs
<b>ProcHours</b>	Number of proc-hours required to complete running jobs
<b>Jobs*</b>	Number of jobs completed
<b>%</b>	Percentage of total jobs that were completed by account
<b>PHReq*</b>	Total proc-hours requested by completed jobs
<b>%</b>	Percentage of total proc-hours requested by completed jobs that were requested by account

Column	Description
<b>PHDed</b>	Total proc-hours dedicated to active and completed jobs. The proc-hours dedicated to a job are calculated by multiplying the number of allocated procs by the length of time the procs were allocated, regardless of the job's CPU usage.
<b>%</b>	Percentage of total proc-hours dedicated that were dedicated by account
<b>FSTgt</b>	Fairshare target. An account's fairshare target is specified in the <code>fs.cfg</code> file. This value should be compared to the account's node-hour dedicated percentage to determine if the target is being met.
<b>AvgXF*</b>	Average expansion factor for jobs completed. A job's XFactor (expansion factor) is calculated by the following formula: $(\text{QueuedTime} + \text{RunTime}) / \text{WallClockLimit}$ .
<b>MaxXF*</b>	Highest expansion factor received by jobs completed
<b>AvgQH*</b>	Average queue time (in hours) of jobs
<b>Effic</b>	Average job efficiency. Job efficiency is calculated by dividing the actual node-hours of CPU time used by the job by the node-hours allocated to the job.
<b>WCAcc*</b>	Average wallclock accuracy for jobs completed. Wallclock accuracy is calculated by dividing a job's actual run time by its specified wallclock limit. <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p> A job's wallclock accuracy is capped at 100% so even if a job exceeds its requested walltime it will report an accuracy of 100%.</p> </div>

\* These fields are empty until an account has completed at least one job.

Example 3-65: Group statistics

```
> showstats -g
Group Statistics Initialized Tue Aug 26 14:32:39
-----|----- Running -----|----- Completed -----
-----|
GroupName  GID Jobs  Procs ProcHours  Jobs  %  PHReq  %  PHDed  %  FSTgt
AvgXF  MaxXF  AvgQH  Effic  WCAcc
univ  214  16  92  1394.52  229  39.15  18486  45.26  7003.5  41.54  40.00
0.77  8.15  5.21  90.70  34.69
daf  204  11  63  855.27  43  7.35  6028  14.76  3448.4  20.45  6.25
0.71  5.40  3.14  98.64  40.83
dnavy 207  6  72  728.12  90  15.38  5974  14.63  3170.7  18.81  6.25
0.37  4.88  0.52  82.01  24.14
govt  232  3  24  220.72  77  13.16  2537  6.21  1526.6  9.06  -----
1.53  14.81  0.42  98.73  28.40
asp  227  0  0  0.00  12  2.05  6013  14.72  958.6  5.69  2.50
1.78  8.61  5.60  83.64  17.04
derim 229  0  0  0.00  74  12.65  669  1.64  352.5  2.09  -----
0.50  1.93  0.51  96.03  32.60
dchall 274  0  0  0.00  3  0.51  447  1.10  169.2  1.00  25.00
0.52  0.88  2.49  95.82  33.67
nih  239  0  0  0.00  17  2.91  170  0.42  148.1  0.88  -----
0.95  1.83  0.14  97.59  84.31
darmy 205  0  0  0.00  31  5.30  366  0.90  53.9  0.32  6.25
0.14  0.59  0.07  81.33  12.73
systems 80  0  0  0.00  6  1.03  67  0.16  22.4  0.13  -----
4.07  8.49  1.23  28.68  37.34
pdc  252  0  0  0.00  1  0.17  64  0.16  5.1  0.03  -----
10.85 10.85 10.77 27.90  7.40
staff  1  0  0  0.00  1  0.17  12  0.03  0.2  0.00  -----
0.04  0.04  0.19  21.21  1.20
```

*This example shows a statistical listing of all active groups. The top line (Group Statistics Initialized...) of the output indicates the beginning of the timeframe covered by the displayed statistics.*

*The statistical output is divided into two categories, Running and Completed. Running statistics include information about jobs that are currently running. Completed statistics are compiled using historical information from both running and completed jobs.*

The fields are as follows:

Column	Description
<b>GroupName</b>	Name of group.
<b>GID</b>	Group ID of group.
<b>Jobs</b>	Number of running jobs.
<b>Procs</b>	Number of procs allocated to running jobs.
<b>ProcHours</b>	Number of proc hours required to complete running jobs.
<b>Jobs*</b>	Number of jobs completed.

Column	Description
%	Percentage of total jobs that were completed by group.
<b>PHReq*</b>	Total proc-hours requested by completed jobs.
%	Percentage of total proc-hours requested by completed jobs that were requested by group.
<b>PHDed</b>	Total proc-hours dedicated to active and completed jobs. The proc-hours dedicated to a job are calculated by multiplying the number of allocated procs by the length of time the procs were allocated, regardless of the job's CPU usage.
%	Percentage of total proc-hours dedicated that were dedicated by group.
<b>FSTgt</b>	Fairshare target. A group's fairshare target is specified in the <code>fs.cfg</code> file. This value should be compared to the group's node-hour dedicated percentage to determine if the target is being met.
<b>AvgXF*</b>	Average expansion factor for jobs completed. A job's XFactor (expansion factor) is calculated by the following formula: $(\text{QueuedTime} + \text{RunTime}) / \text{WallClockLimit}$ .
<b>MaxXF*</b>	Highest expansion factor received by jobs completed.
<b>AvgQH*</b>	Average queue time (in hours) of jobs.
<b>Effic</b>	Average job efficiency. Job efficiency is calculated by dividing the actual node-hours of CPU time used by the job by the node-hours allocated to the job.
<b>WCAcc*</b>	Average wallclock accuracy for jobs completed. Wallclock accuracy is calculated by dividing a job's actual run time by its specified wallclock limit. <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin-top: 10px;">  A job's wallclock accuracy is capped at 100% so even if a job exceeds its requested walltime it will report an accuracy of 100%. </div>

\* These fields are empty until a group has completed at least one job.

**Example 3-66: Node statistics**

```
> showstats -n
node stats from Mon Jul 10 00:00:00 to Mon Jul 10 16:30:00
node      CfgMem  MinMem  MaxMem  AvgMem  | CfgProcs  MinLoad  MaxLoad  AvgLoad
node01    58368    0    21122  5841    |    32     0.00    32.76    27.62
node02   122880    0    19466   220    |    30     0.00    33.98    29.54
node03   18432    0    9533   2135   |    24     0.00    25.10    18.64
node04   60440    0   17531  4468    |    32     0.00    30.55    24.61
node05   13312    0    2597   1189    |     8     0.00     9.85     8.45
node06   13312    0    3800   1112    |     8     0.00     8.66     5.27
node07   13312    0    2179   1210    |     8     0.00     9.62     8.27
node08   13312    0    3243   1995    |     8     0.00    11.71     8.02
node09   13312    0    2287   1943    |     8     0.00    10.26     7.58
node10   13312    0    2183   1505    |     8     0.00    13.12     9.28
node11   13312    0    3269   2448    |     8     0.00     8.93     6.71
node12   13312    0   10114  6900    |     8     0.00    13.13     8.44
node13   13312    0    2616   2501    |     8     0.00     9.24     8.21
node14   13312    0    3888    869    |     8     0.00     8.10     3.85
node15   13312    0    3788    308    |     8     0.00     8.40     4.67
node16   13312    0    4386   2191    |     7     0.00    18.37     8.36
node17   13312    0    3158   1870    |     8     0.00     8.95     5.91
node18   13312    0    5022   2397    |     8     0.00    19.25     8.19
node19   13312    0    2437   1371    |     8     0.00     8.98     7.09
node20   13312    0    4474   2486    |     8     0.00     8.51     7.11
node21   13312    0    4111   2056    |     8     0.00     8.93     6.68
node22   13312    0    5136   2313    |     8     0.00     8.61     5.75
node23   13312    0    1850   1752    |     8     0.00     8.39     5.71
node24   13312    0    3850   2539    |     8     0.00     8.94     7.80
node25   13312    0    3789   3702    |     8     0.00    21.22    12.83
node26   13312    0    3809   1653    |     8     0.00     9.34     4.91
node27   13312    0    5637     70    |     4     0.00    17.97     2.46
node28   13312    0    3076   2864    |     8     0.00    22.91    10.33
```

**Example 3-67: Verbose statistics**

```
> showstats -v
current scheduler time: Sat Aug 18 18:23:02 2007
moab active for      00:00:01  started on Wed Dec 31 17:00:00
statistics for iteration      0  initialized on Sat Aug 11 23:55:25
Eligible/Idle Jobs:          6/8   (75.000%)
Active Jobs:                  13
Successful/Completed Jobs:   167/167 (100.000%)
Preempt Jobs:                 0
Avg/Max QTime (Hours):       0.34/2.07
Avg/Max XFactor:             1.165/3.26
Avg/Max Bypass:              0.40/8.00
Dedicated/Total ProcHours:   4.46K/4.47K (99.789%)
Preempt/Dedicated ProcHours: 0.00/4.46K (0.000%)
Current Active/Total Procs:  32/32  (100.0%)
Current Active/Total Nodes:  16/16  (100.0%)
Avg WallClock Accuracy:     64.919%
Avg Job Proc Efficiency:     99.683%
Min System Utilization:     87.323% (on iteration 46)
Est/Avg Backlog:            02:14:06/03:02:567
```

*This example shows a concise summary of the system scheduling state. Note that `showstats` and `showstats-s` are equivalent.*

*The first line of output indicates the number of scheduling iterations performed by the current scheduling process, followed by the time the scheduler started. The second line indicates the amount of time the Moab Scheduler has been scheduling in HH:MM:SS notation followed by the statistics initialization time.*

The fields are as follows:

Column	Description
<b>Active Jobs</b>	Number of jobs currently active (Running or Starting).
<b>Eligible Jobs</b>	Number of jobs in the system queue (jobs that are considered when scheduling).
<b>Idle Jobs</b>	Number of jobs both in and out of the system queue that are in the LoadLeveler Idle state.
<b>Completed Jobs</b>	Number of jobs completed since statistics were initialized.
<b>Successful Jobs</b>	Jobs that completed successfully without abnormal termination.
<b>XFactor</b>	Average expansion factor of all completed jobs.
<b>Max XFactor</b>	Maximum expansion factor of completed jobs.
<b>Max Bypass</b>	Maximum bypass of completed jobs.
<b>Available ProcHours</b>	Total proc-hours available to the scheduler.
<b>Dedicated ProcHours</b>	Total proc-hours made available to jobs.
<b>Effic</b>	Scheduling efficiency (DedicatedProcHours / Available ProcHours).
<b>Min Efficiency</b>	Minimum scheduling efficiency obtained since scheduler was started.
<b>Iteration</b>	Iteration on which the minimum scheduling efficiency occurred.
<b>Available Procs</b>	Number of procs currently available.
<b>Busy Procs</b>	Number of procs currently busy.
<b>Effic</b>	Current system efficiency (BusyProcs/AvailableProcs).
<b>WallClock Accuracy</b>	Average wallclock accuracy of completed jobs (job-weighted average).
<b>Job Efficiency</b>	Average job efficiency (UtilizedTime / DedicatedTime).

Column	Description
<b>Est Backlog</b>	Estimated backlog of queued work in hours.
<b>Avg Backlog</b>	Average backlog of queued work in hours.

**Example 3-68: User statistics**

```

> showstats -u
User Statistics Initialized Tue Aug 26 14:32:39
      |----- Running -----|----- Completed -----
-----|-----
  UserName  UID  Jobs  Procs  ProcHours  Jobs   %   PHReq   %   PHDed   %   FSTgt
AvgXF  MaxXF  AvgQH  Effic  WCAcc
moorejtc 2617   1    16     58.80     2   0.34   221   0.54  1896.6  11.25  -----
1.02   1.04   0.14  99.52  100.00
zhong    1767   3    24    220.72    20   3.42  2306   5.65  1511.3   8.96  -----
0.71   0.96   0.49  99.37   67.48
lui      2467   0     0     0.00     16   2.74  1970   4.82  1505.1   8.93  -----
1.02   6.33   0.25  98.96   57.72
evans    3092   0     0     0.00     62  10.60  4960  12.14  1464.3   8.69   5.0
0.62   1.64   5.04  87.64   30.62
wengel   2430   2    64    824.90     1   0.17   767   1.88   630.3   3.74  -----
0.18   0.18   4.26  99.63    0.40
mukho    2961   2    16    71.06     6   1.03   776   1.90   563.5   3.34  -----
0.31   0.82   0.20  93.15   30.28
jimenez  1449   1    16    302.29     2   0.34   768   1.88   458.3   2.72  -----
0.80   0.98   2.31  97.99   70.30
neff     3194   0     0     0.00     74  12.65  669   1.64   352.5   2.09  10.0
0.50   1.93   0.51  96.03   32.60
cholik   1303   0     0     0.00     2   0.34   552   1.35   281.9   1.67  -----
1.72   3.07  25.35  99.69   66.70
jshoemak 2508   1    24    572.22     1   0.17   576   1.41   229.1   1.36  -----
0.55   0.55   3.74  99.20   39.20
kudo     2324   1     8    163.35     6   1.03  1152   2.82   211.1   1.25  -----
0.12   0.34   1.54  96.77    5.67
xztang   1835   1     8    18.99     ---  ---  ---  ---  176.3   1.05  10.0  -----
-  -----  99.62  -----
feller   1880   0     0     0.00     17   2.91   170   0.42   148.1   0.88  -----
0.95   1.83   0.14  97.59   84.31
maxia    2936   0     0     0.00     1   0.17   191   0.47   129.1   0.77   7.5
0.88   0.88   4.49  99.84   69.10
ktgnov71 2838   0     0     0.00     1   0.17   192   0.47    95.5   0.57  -----
0.53   0.53   0.34  90.07   51.20

```

*This example shows a statistical listing of all active users. The top line (User Statistics Initialized...) of the output indicates the timeframe covered by the displayed statistics.*

*The statistical output is divided into two statistics categories, Running and Completed. Running statistics include information about jobs that are currently running. Completed statistics are compiled using historical information from both running and completed jobs.*

The fields are as follows:

Column	Description
<b>UserName</b>	Name of user.
<b>UID</b>	User ID of user.
<b>Jobs</b>	Number of running jobs.
<b>Procs</b>	Number of procs allocated to running jobs.
<b>ProcHours</b>	Number of proc-hours required to complete running jobs.
<b>Jobs*</b>	Number of jobs completed.
<b>%</b>	Percentage of total jobs that were completed by user.
<b>PHReq*</b>	Total proc-hours requested by completed jobs.
<b>%</b>	Percentage of total proc-hours requested by completed jobs that were requested by user.
<b>PHDed</b>	Total proc-hours dedicated to active and completed jobs. The proc-hours dedicated to a job are calculated by multiplying the number of allocated procs by the length of time the procs were allocated, regardless of the job's CPU usage.
<b>%</b>	Percentage of total proc-hours dedicated that were dedicated by user.
<b>FSTgt</b>	Fairshare target. A user's fairshare target is specified in the <code>fs.cfg</code> file. This value should be compared to the user's node-hour dedicated percentage to determine if the target is being met.
<b>AvgXF*</b>	Average expansion factor for jobs completed. A job's XFactor (expansion factor) is calculated by the following formula: $(\text{QueuedTime} + \text{RunTime}) / \text{WallClockLimit}$ .
<b>MaxXF*</b>	Highest expansion factor received by jobs completed.
<b>AvgQH*</b>	Average queue time (in hours) of jobs.
<b>Effic</b>	Average job efficiency. Job efficiency is calculated by dividing the actual node-hours of CPU time used by the job by the node-hours allocated to the job.

Column	Description
<b>WCacc*</b>	<p>Average wallclock accuracy for jobs completed. Wallclock accuracy is calculated by dividing a job's actual run time by its specified wallclock limit.</p> <div data-bbox="410 390 1433 478" style="border: 1px solid #ccc; padding: 5px;"><p> A job's wallclock accuracy is capped at 100% so even if a job exceeds its requested walltime it will report an accuracy of 100%.</p></div>

\* These fields are empty until a user has completed at least one job.

Example 3-69: Fairshare tree statistics

```

> showstats -T
statistics initialized Mon Jul 10 15:29:41
|----- Active -----|----- Completed -----
|-----|
user      Jobs Procs ProcHours  Mem Jobs   %   PHReq   %   PHDed   %   FSTgt
  AvgXF  MaxXF  AvgQH  Effic  WCAcc
root      0      0      0.00    0    56 100.00  2.47K 100.00  1.58K  48.87 -----
  1.22   0.00   0.24 100.00  58.84
11.1      0      0      0.00    0    25  44.64  845.77 34.31  730.25 22.54 -----
  1.97   0.00   0.20 100.00  65.50
Administrati 0      0      0.00    0    10  17.86  433.57 17.59  197.17  6.09 -----
  3.67   0.00   0.25 100.00  62.74
Engineering  0      0      0.00    0    15  26.79  412.20 16.72  533.08 16.45 -----
  0.83   0.00   0.17 100.00  67.35
11.2      0      0      0.00    0    31  55.36  1.62K 65.69  853.00 26.33 -----
  0.62   0.00   0.27 100.00  53.46
Shared     0      0      0.00    0     3   5.36   97.17   3.94   44.92   1.39 -----
  0.58   0.00   0.56 100.00  31.73
Test       0      0      0.00    0     3   5.36   14.44   0.59   14.58   0.45 -----
  0.43   0.00   0.17 100.00  30.57
Research   0      0      0.00    0    25  44.64  1.51K 61.16  793.50 24.49 -----
  0.65   0.00   0.24 100.00  58.82

> showstats -T 2
statistics initialized Mon Jul 10 15:29:41
|----- Active -----|----- Completed -----
|-----|
user      Jobs Procs ProcHours  Mem Jobs   %   PHReq   %   PHDed   %   FSTgt
  AvgXF  MaxXF  AvgQH  Effic  WCAcc
Test      0      0      0.00    0    22   4.99  271.27  0.55  167.42  0.19 -----
  3.86   0.00   2.89 100.00  60.76
Shared    0      0      0.00    0    59  13.38 12.30K 24.75  4.46K   5.16 -----
  6.24   0.00  10.73 100.00  49.87
Research  0      0      0.00    0   140 31.75  9.54K 19.19  5.40K   6.25 -----
  2.84   0.00   5.52 100.00  57.86
Administrati 0      0      0.00    0    84 19.05  7.94K 15.96  4.24K   4.91 -----
  4.77   0.00   0.34 100.00  62.31
Engineering 0      0      0.00    0   136 30.84 19.67K 39.56 28.77K 33.27 -----
  3.01   0.00   3.66 100.00  63.70

> showstats -T 11.1
statistics initialized Mon Jul 10 15:29:41
|----- Active -----|----- Completed -----
|-----|
user      Jobs Procs ProcHours  Mem Jobs   %   PHReq   %   PHDed   %   FSTgt
  AvgXF  MaxXF  AvgQH  Effic  WCAcc
11.1      0      0      0.00    0   220 49.89 27.60K 55.52 33.01K 38.17 -----
  3.68   0.00   2.39 100.00  63.17
Administrati 0      0      0.00    0    84 19.05  7.94K 15.96  4.24K   4.91 -----
  4.77   0.00   0.34 100.00  62.31
Engineering  0      0      0.00    0   136 30.84 19.67K 39.56 28.77K 33.27 -----
  3.01   0.00   3.66 100.00  63.70

```

**Example 3-70: Statistics from an absolute time frame**

```
> showstats -c batch -v -t 00:00:01_01/01/13,23:59:59_12/31/13
statistics initialized Wed Jan 1 00:00:00

----- Active ----- Completed -----
-----

class Jobs Procs ProcHours Mem Jobs      %   PHReq      %   PHDed      %   FSTgt AvgXF
MaxXF  AvgQH  Effic  WCAcc
batch  0      0      0.00   0   23  100.00   15  100.00    1  100.00  -----  0.40
5.01   0.00  88.94  39.87
```

*Moab returns information about the class `batch` from January 1, 2013 to December 31, 2013. For more information about specifying absolute dates, see "Absolute Time Format" in [TIMESPEC on page 362](#).*

**Example 3-71: Statistics from a relative time frame**

```
> showstats -u bob -v -t -30:00:00:00
statistics initialized Mon Nov 11 15:30:00

----- Active ----- Completed -----
-----

user Jobs Procs ProcHours Mem Jobs      %   PHReq      %   PHDed      %   FSTgt AvgXF
MaxXF  AvgQH  Effic  WCAcc
bob    0      0      0.00   0   23  100.00   15  100.00    1  100.00  -----  0.40
5.01   0.00  88.94  39.87
```

*Moab returns information about user `bob` from the past 30 days. For more information about specifying relative dates, see "Relative Time Format" in [TIMESPEC on page 362](#).*

**Related topics**

- [Moab Client Installation](#) - explains how to distribute this command to client nodes
- [mschedctl -f](#) command - re-initialize statistics
- [showstats -f](#) command - display full matrix statistics

**showstats -f**

**Synopsis**

`showstats -f <statistictype>`

**Overview**

Shows table of various scheduler statistics.

This command displays a table of the selected Moab Scheduler statistics, such as expansion factor, bypass count, jobs, proc-hours, wallclock accuracy, and backfill information.

**i** Statistics are aggregated over time. This means statistical information is not available for time frames and the `-t` option is not supported with `showstats -f`.

## Access

This command can be run by any Moab Scheduler Administrator.

## Parameters

Parameter	Description
<b>AVGBYPASS</b>	Average bypass count. Includes summary of job-weighted expansion bypass and total samples.
<b>AVGQTIME</b>	Average queue time. Includes summary of job-weighted queue time and total samples.
<b>AVGXFACTOR</b>	Average expansion factor. Includes summary of job-weighted expansion factor, processor-weighted expansion factor, processor-hour-weighted expansion factor, and total number of samples.
<b>BFCOUNT</b>	Number of jobs backfilled. Includes summary of job-weighted backfill job percent and total samples.
<b>BFPHRUN</b>	Number of proc-hours backfilled. Includes summary of job-weighted backfill proc-hour percentage and total samples.
<b>ESTSTARTTIME</b>	Job start time estimate for jobs meeting specified processor/duration criteria. This estimate is based on the <a href="#">reservation start time analysis</a> algorithm.
<b>JOB COUNT</b>	Number of jobs. Includes summary of total jobs and total samples.
<b>MAXBYPASS</b>	Maximum bypass count. Includes summary of overall maximum bypass and total samples.
<b>MAXXFACTOR</b>	Maximum expansion factor. Includes summary of overall maximum expansion factor and total samples.
<b>PHREQUEST</b>	proc-hours requested. Includes summary of total proc-hours requested and total samples.
<b>PHRUN</b>	proc-hours run. Includes summary of total proc-hours run and total samples.
<b>QOSDELIVERED</b>	Quality of service delivered. Includes summary of job-weighted quality of service success rate and total samples.
<b>WCACCURACY</b>	Wallclock accuracy. Includes summary of overall wall clock accuracy and total samples.

## Examples

### Example 3-72:

```

> showstats -f AVGXFACTOR
Average XFactor Grid
[ NODES ][ 00:02:00 ][ 00:04:00 ][ 00:08:00 ][ 00:16:00 ][ 00:32:00 ][ 01:04:00 ][
02:08:00 ][ 04:16:00 ][ 08:32:00 ][ 17:04:00 ][ 34:08:00 ][ TOTAL ]
[ 1 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ 2 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ 4 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
1.00 1][ ----- ][ 1.12 2][ ----- ][ ----- ][ 1.10 3]
[ 8 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
1.00 2][ 1.24 2][ ----- ][ ----- ][ ----- ][ 1.15 4]
[ 16 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ 1.01 2]
[ 32 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ 64 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ 128 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ 256 ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
[ T TOT ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ----- ][ ---
1.00 3][ 1.24 2][ 1.12 2][ ----- ][ ----- ][ ----- ][ ---
Job Weighted X Factor: 1.0888
Node Weighted X Factor: 1.1147
NS Weighted X Factor: 1.1900
Total Samples: 9

```

The `showstats -f` command returns a table with data for the specified `STATISTICTYPE` parameter. The left-most column shows the maximum number of processors required by the jobs shown in the other columns. The column headers indicate the maximum wallclock time (in HH:MM:SS notation) requested by the jobs shown in the columns. The data returned in the table varies by the `STATISTICTYPE` requested. For table entries with one number, it is of the data requested. For table entries with two numbers, the left number is the data requested and the right number is the number of jobs used to calculate the average. Table entries that contain only dashes (-----) indicate no job has completed that matches the profile associated for this inquiry. The bottom row shows the totals for each column. Following each table is a summary, which varies by the `STATISTICTYPE` requested.

 The column and row break down can be adjusted using the `STATPROC*` and `STATTIME*` parameters respectively.

This particular example shows the average expansion factor grid. Each table entry indicates two pieces of information — the average expansion factor for all jobs that meet this slot's profile and the number of jobs that were used to calculate this average. For example, the XFactors of two jobs were averaged to obtain an average XFactor of 1.24 for jobs requiring over 2 hours 8 minutes, but not more than 4 hours 16 minutes and between 5 and 8 processors. Totals along the bottom provide overall XFactor averages weighted by job, processors, and processor-hours.

### Related topics

- Moab Client Installation - explains how to distribute this command to client nodes
- [mschedctl -f](#) command
- [showstats](#) command
- [STATPROCMIN](#) parameter
- [STATPROCSTEP](#) parameter
- [STATPROCSTEP](#) parameter
- [STATPROCSTEP](#) parameter
- [STATTIMEMIN](#) parameter

- [STATTIMESTEPCOUNT](#) parameter
- [STATTIMESTEPSIZE](#) parameter

## TIMESPEC

### Relative Time Format

The relative time format specifies a time by using the current time as a reference and specifying a time offset.

#### Format

+[[[DD:]HH:]MM:]SS

#### Examples

2 days, 3 hours and 57 seconds in the future:

```
+02:03:0:57
```

21 days (3 weeks) in the future:

```
+21:0:0:0
```

30 seconds in the future:

```
+30
```

### Absolute Time Format

The absolute time format specifies a specific time in the future.

#### Format

[HH[:MM[:SS]]][\_MO[/DD[/YY]]] i.e. 14:30\_06/20

#### Examples

1 PM, March 1 (this year)

```
13:00_03/01
```

### Deprecated commands

#### canceljob



This command is deprecated. Use [mjobctl -c](#) instead.

### Synopsis

canceljob *jobid* [*jobid*]...

## Overview

The `canceljob` command is used to selectively cancel the specified job(s) (active, idle, or non-queued) from the queue.

## Access

This command can be run by any Moab Administrator and by the owner of the job (see [ADMINCFG](#)).

Flag	Name	Format	Default	Description	Example
<b>-h</b>	HELP		N/A	Display usage information	<pre>&gt; canceljob -h</pre>
	JOB ID	<STRING>	---	a jobid, a job expression, or the keyword ALL	<pre>&gt; canceljob 13001 13003</pre>

## Examples

*Example 3-73: Cancel job 6397*

```
> canceljob 6397
```

### changeparam



This command is deprecated. Use [mschedctl -m](#) instead.

## Synopsis

`changeparam` [parametervalue](#)

## Overview

The `changeparam` command is used to dynamically change the value of any parameter which can be specified in the `moab.cfg` file. The changes take effect at the beginning of the next scheduling iteration. They are not persistent, only lasting until Moab is shut down.

`changeparam` is a compact command of [mschedctl -m](#).

## Access

This command can be run by a level 1 Moab administrator.

### diagnose



This command is deprecated. Use [mdiag](#) instead.

## Synopsis

[diagnose -a](#) [*accountid*]  
[diagnose -b](#) [-l *policylevel*] [-t *partition*]  
[diagnose -c](#) [*classid*]  
[diagnose -C](#) [*configfile*]  
[diagnose -f](#) [-o user|group|account|qos|class]  
[diagnose -g](#) [*groupid*]  
[diagnose -j](#) [*jobid*]  
[diagnose -L](#)  
[diagnose -m](#) [*rackid*]  
[diagnose -n](#) [-t *partition*] [*nodeid*]  
[diagnose -p](#) [-t *partition*]  
[diagnose -q](#) [*qosid*]  
[diagnose -r](#) [*reservationid*]  
[diagnose -R](#) [*resourcemanagername*]  
[diagnose -s](#) [*standingreservationid*]  
[diagnose -S](#) [diagnose -u](#) [*userid*]  
[diagnose -v](#)  
[diagnose -x](#)

## Overview

The `diagnose` command is used to display information about various aspects of scheduling and the results of internal diagnostic tests.

### releasehold



This command is deprecated. Use [mjobctl -u](#) instead.

## Synopsis

`releasehold` [-a|-b] [jobexp](#)

## Overview

Release hold on specified job(s).

This command allows you to release batch holds or all holds (system, user, and batch) on specified jobs. Any number of jobs may be released with this command.

## Access

By default, this command can be run by any Moab Scheduler Administrator.

## Parameters

JOBEXP	Job expression of job(s) to release.
--------	--------------------------------------

## Flags

-a	Release all types of holds (user, system, batch) for specified job(s).
-b	Release batch hold from specified job(s).
-h	Help for this command.

## Examples

### Example 3-74: `releasehold -b`

```
> releasehold -b 6443
batch hold released for job 6443
```

*In this example, a batch hold was released from this one job.*

### Example 3-75: `releasehold -a`

```
> releasehold -a "81[1-6]"
holds modified for job 811
holds modified for job 812
holds modified for job 813
holds modified for job 814
holds modified for job 815
holds modified for job 816
```

*In this example, all holds were released from the specified jobs.*

## Related topics

- [sethold](#)
- [mjobctl](#)

**releaseres**

 This command is deprecated. Use [mrsvctl -r](#) instead.

**Synopsis**

releaseres [*arguments*] *reservationid* [*reservationid...*]

**Overview**

Release existing reservation.

This command allows Moab Scheduler Administrators to release any user, group, account, job, or system reservation. Users are allowed to release reservations on jobs they own. Note that releasing a reservation on an active job has no effect since the reservation will be automatically recreated.

**Access**

Users can use this command to release any reservation they own. Level 1 and level 2 Moab administrators may use this command to release any reservation.

**Parameters**

RESERVATION ID	Name of reservation to release.
----------------	---------------------------------

**Examples**

*Example 3-76: Release two existing reservations*

```
> releaseres system.1 bob.2
released User reservation 'system.1'
released User reservation 'bob.2'
```

**resetstats**

 This command is deprecated. Use [mschedctl -f](#) instead.

**Synopsis**

resetstats

**Overview**

This command resets all internally-stored Moab Scheduler statistics to the initial start-up state as of the time the command was executed.

## Access

By default, this command can be run by level 1 scheduler administrators.

## Examples

Example 3-77:

```
> resetstats Statistics Reset at time Wed Feb 25 23:24:55 2011
```

Related topics

- Moab Client Installation - explains how to distribute this command to client nodes

### runjob



This command is deprecated. Use [mjobctl -x](#) instead.

## Synopsis

```
runjob [-c|-f|-n nodelist|-p partition|-s|-x] jobid
```

## Overview

This command will attempt to immediately start the specified job.

`runjob` is a deprecated command, replaced by [mjobctl](#).

## Access

By default, this command can be run by any Moab administrator.

## Parameters

JOBID	Name of the job to run.
-------	-------------------------

Args	Description
<b>-c</b>	Clear job parameters from previous runs (used to clear PBS neednodes attribute after PBS job launch failure)
<b>-f</b>	Attempt to <i>force</i> the job to run, ignoring throttling policies
<b>-n</b> <b>&lt;NODELIST&gt;</b>	Attempt to start the job using the specified <i>nodelist</i> where nodenames are comma or colon delimited

Args	Description
<b>-p</b> <b>&lt;PARTITION&gt;</b>	Attempt to start the job in the specified <i>partition</i>
<b>-s</b>	Attempt to <i>suspend</i> the job
<b>-x</b>	Attempt to force the job to run, ignoring throttling policies, QoS constraints, and reservations

## Examples

Example 3-78: Run job *cluster.231*

```
> runjob cluster.231
job cluster.231 successfully started
```

## See Also

- [mjobctl](#)
- [canceljob](#) - cancel a job.
- [checkjob](#) - show detailed status of a job.
- [showq](#) - list queued jobs.

### sethold



This command is deprecated. Use [mjobctl -h](#) instead.

## Synopsis

```
sethold [-b] jobid [jobid...]
```

## Overview

Set hold on specified job(s).

## Permissions

This command can be run by any Moab Scheduler Administrator.

## Parameters

JOB	Job number of job to hold.
-----	----------------------------

## Flags

<b>-b</b>	Set a batch hold. Typically, only the scheduler places batch holds. This flag allows an administrator to manually set a batch hold.
<b>-h</b>	Help for this command.

## Examples

Example 3-79:

```
> sethold -b fr17n02.1072.0 fr15n03.1017.0
Batch Hold Placed on All Specified Jobs
```

*In this example, a batch hold is placed on job fr17n02.1072.0 and job fr15n03.1017.0.*

### setqos



This command is deprecated. Use [mjobctl -m](#) instead.

## Synopsis

```
setqos qosid jobid
```

## Overview

Set Quality Of Service for a specified job.

This command allows users to change the QOS of their own jobs.

## Access

This command can be run by any user.

## Parameters

JOBID	Job name.
QOSID	QOS name.

## Examples

Example 3-80:

```
> setqos high_priority moab.3
```

```
Job QOS Adjusted
```

*This example sets the Quality Of Service to a value of `high_priority` for job `moab.3`.*

### setres



This command is deprecated. Use [mrsvctl -c](#) instead.

## Synopsis

setres [*arguments*] *resourceexpression*

```
[ -a <ACCOUNT_LIST> ]
[ -b <SUBTYPE> ]
[ -c <CHARGE_SPEC> ]
[ -d <DURATION> ]
[ -e <ENDTIME> ]
[ -E ] // EXCLUSIVE
[ -f <FEATURE_LIST> ]
[ -g <GROUP_LIST> ]
[ -n <NAME> ]
[ -o <OWNER> ]
[ -p <PARTITION> ]
[ -q <QUEUE_LIST> ] // (i.e. CLASS_LIST)
[ -Q <QOSLIST> ]
[ -r <RESOURCE_DESCRIPTION> ]
[ -R <RESERVATION_PROFILE> ]
[ -s <STARTTIME> ]
[ -T <TRIGGER> ]
[ -u <USER_LIST> ]
[ -x <FLAGS> ]
```

## Overview

Reserve resources for use by jobs with particular credentials or attributes.

## Access

This command can be run by level 1 and level 2 Moab administrators.

## Parameters

Name	Format	Default	Description
<b>ACCOUNT_LIST</b>	<STRING> [:<STRING>]...	---	List of accounts that will be allowed access to the reserved resources
<b>SUBTYPE</b>	<STRING>	---	Specify the subtype for a reservation
<b>CHARGE_SPEC</b>	<ACCOUNT> [,<GROUP> [,<USER>]]	---	Specifies which credentials will be accountable for unused resources dedicated to the reservation
<b>CLASS_LIST</b>	<STRING> [:<STRING>]...	---	List of classes that will be allowed access to the reserved resource
<b>DURATION</b>	[[[DD:]HH:]MM:]SS	INFINITY	Duration of the reservation (not needed if ENDTIME is specified)
<b>ENDTIME</b>	[HH[:MM[:SS]]][_MO[/DD[/YY]]] or +[[[DD:]HH:]MM:]SS	INFINITY	Absolute or relative time reservation will end (not required if Duration specified)
<b>EXCLUSIVE</b>	N/A	N/A	Requests exclusive access to resources
<b>FEATURE_LIST</b>	<STRING> [:<STRING>]...	---	List of node features which must be possessed by the reserved resources
<b>FLAGS</b>	<STRING> [:<STRING>]...	---	List of reservation flags (See <a href="#">Managing Reservations</a> for details)
<b>GROUP_LIST</b>	<STRING> [:<STRING>]...	---	List of groups that will be allowed access to the reserved resources
<b>NAME</b>	<STRING>	Name set to first name listed in ACL or SYSTEM if no ACL specified	Name for new reservation

Name	Format	Default	Description
<b>OWNER</b>	<CREDTYPE> :<CREDID> where CREDTYPE is one of user, group, acct, class, or qos	N/A	Specifies which credential is granted reservation ownership privileges
<b>PARTITION</b>	<STRING>	[ANY]	Partition in which resources must be located
<b>QOS_LIST</b>	<STRING> [:<STRING>]...	---	List of QOS's that will be allowed access to the reserved resource
<b>RESERVATION_ PROFILE</b>	Existing reservation profile ID	N/A	Requests that default reservation attributes be loaded from the specified reservation profile (see <a href="#">RSVPROFILE</a> )
<b>RESOURCE_ DESCRIPTION</b>	Colon delimited list of zero or more of the following <ATTR>=<VALUE> pairs PROCS=<INTEGER> MEM=<INTEGER> DISK=<INTEGER> SWAP=<INTEGER> GRES=<STRING>	PROCS=-1	Specifies the resources to be reserved per task. (-1 indicates all resources on node)
<b>RESOURCE_ EXPRESSION</b>	ALL or TASKS{== >=} <TASKCOUNT> or <HOST_REGEX>	Required Field. No Default	Specifies the tasks to reserve. ALL indicates all resources available should be reserved. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"><b>i</b> If ALL or a host expression is specified, Moab will apply the reservation regardless of existing reservations and exclusive issues. If TASKS is used, Moab will only allocate accessible resources.</div>
<b>STARTTIME</b>	[HH[:MM[:SS]]]_ MO[/DD[/YY]]] or +[[[DD:]HH:]MM:] SS	NOW	Absolute or relative time reservation will start

Name	Format	Default	Description
<b>TRIGGER</b>	<STRING>	N/A	Comma delimited reservation trigger list following format described in the <a href="#">trigger format</a> section of the reservation configuration overview.
<b>USER_LIST</b>	<STRING> [:<STRING>]...	---	List of users that will be allowed access to the reserved resources

## Description

The `setres` command allows an arbitrary block of resources to be reserved for use by jobs which meet the specified access constraints. The timeframe covered by the reservation can be specified on either an absolute or relative basis. Only jobs with credentials listed in the reservation ACL (i.e., **USERLIST**, **GROUPLIST**,...) can utilize the reserved resources. However, these jobs still have the freedom to utilize resources outside of the reservation. The reservation will be assigned a name derived from the ACL specified. If no reservation ACL is specified, the reservation is created as a system reservation and no jobs will be allowed access to the resources during the specified timeframe (valuable for system maintenance, etc.). See the [Reservation Overview](#) for more information.

Reservations can be viewed using the [showres](#) command and can be released using the [releaseres](#) command.

## Examples

### Example 3-81:

```
> setres -u john:mary -s +24:00:00 -d 8:00:00 TASKS==2
reservation 'john.1' created on 2 nodes (2 tasks)
node001:1
node005:1
```

*Reserve two nodes for use by users john and mary for a period of 8 hours starting in 24 hours.*

### Example 3-82:

```
> setres -s 8:00:00_06/20 -e 17:00:00_06/22 ALL
reservation 'system.1' created on 8 nodes (8 tasks)
node001:1
node002:1
node003:1
node004:1
node005:1
node006:1
node007:1
node008:1
```

*Schedule a system wide reservation to allow system maintenance on Jun 20, 8:00 AM until Jun 22, 5:00 PM.*

**Example 3-83:**

```
> setres -r PROCS=1:MEM=512 -g staff -l interactive 'node00[3-6] '
reservation 'staff.1' created on 4 nodes (4 tasks)
node003:1
node004:1
node005:1
node006:1
```

*Reserve one processor and 512 MB of memory on nodes node003 through node node006 for members of the group staff and jobs in the interactive class.*

**setspri**

 This command is deprecated. Use [mjobctl -p](#) instead.

**Synopsis**

setspri [-r] priorityjobid

**Overview**

(This command is deprecated by the [mjobctl command](#))

Set or remove absolute or relative system priorities for a specified job.

This command allows you to set or remove a system priority level for a specified job. Any job with a system priority level set is guaranteed a higher priority than jobs without a system priority. Jobs with higher system priority settings have priority over jobs with lower system priority settings.

**Access**

This command can be run by any Moab Scheduler Administrator.

**Parameters**

JOB	Name of job.
PRIORITY	System priority level. By default, this priority is an absolute priority overriding the policy generated priority value. Range is 0 to clear, 1 for lowest, 1000 for highest. The given value is added onto the system priority (see 32-bit and 64-bit values below), except for a given value of zero. If the '-r' flag is specified, the system priority is relative, adding or subtracting the specified value from the policy generated priority.  If a relative priority is specified, any value in the range +/- 1,000,000,000 is acceptable.

**Flags**

-r	Set relative system priority on job.
----	--------------------------------------

## Examples

### Example 3-84:

```
> setspri 10 orion.4752
job system priority adjusted
```

*In this example, a system priority of 10 is set for job orion.4752.*

### Example 3-85:

```
> setspri 0 clusterB.1102
job system priority adjusted
```

*In this example, system priority is cleared for job clusterB.1102.*

### Example 3-86:

```
> setspri -r 100000 job.00001
job system priority adjusted
```

*In this example, the job's priority will be increased by 100000 over the value determine by configured priority policy.*

 This command is deprecated. Use [mjobctl](#) instead.

### showconfig

 This command is deprecated. Use [mschedctl -l](#) instead.

## Synopsis

```
showconfig [-v]
```

## Overview

View the current configurable parameters of the Moab Scheduler.

The `showconfig` command shows the current scheduler version and the settings of all "in memory" parameters. These parameters are set via internal defaults, command line arguments, environment variable settings, parameters in the `moab.cfg` file, and via the [mschedctl -m](#) command. Because of the many sources of configuration settings, the output may differ from the contents of the `moab.cfg` file. The output is such that it can be saved and used as the contents of the `moab.cfg` file if desired.

## Access

This command can be run by a level 1, 2, or 3 Moab administrator.

## Flags

- <b>h</b>	Help for this command.
- <b>v</b>	This optional flag turns on verbose mode, which shows all possible Moab Scheduler parameters and their current settings. If this flag is not used, this command operates in context-sensitive terse mode, which shows only relevant parameter settings.

## Examples

### Example 3-87: showconfig

```
> showconfig
# moab scheduler version 4.2.4 (PID: 11080)
BACKFILLPOLICY          FIRSTFIT
BACKFILLMETRIC          NODES
ALLOCATIONPOLICY         MINRESOURCE
RESERVATIONPOLICY      CURRENTHIGHEST
...
```

**i** The `showconfig` command without the `-v` flag does not show the settings of all parameters. It does show all major parameters and all parameters which are in effect and have been set to non-default values. However, it hides other rarely used parameters and those which currently have no effect or are set to default values. To show the settings of all parameters, use the `-v` (verbose) flag. This will provide an extended output. This output is often best used in conjunction with the `grep` command as the output can be voluminous.

### Related topics

- Use the [mschedctl -m](#) command to change the various Moab Scheduler parameters.
- See the [Parameters](#) document for details about configurable parameters.

# Prioritizing Jobs and Allocating Resources

- [Job Prioritization](#) on page 377
- [Node Allocation Policies](#) on page 393
- [Node Access Policies](#) on page 403
- [Node Availability Policies](#) on page 404

## Job Prioritization

In general, prioritization is the process of determining which of many options best fulfills overall goals. In the case of scheduling, a site will often have multiple, independent goals that may include maximizing system utilization, giving preference to users in specific projects, or making certain that no job sits in the queue for more than a given period of time. The approach used by Moab in representing a multi-faceted set of site goals is to assign weights to the various objectives so an overall value or priority can be associated with each potential scheduling decision. With the jobs prioritized, the scheduler can roughly fulfill site objectives by starting the jobs in priority order.

- [Priority Overview](#)
- [Job Priority Factors](#)
- [Fairshare Job Priority Example on page 388](#)
- [Common Priority Usage](#)
- [Prioritization Strategies](#)
- [Manual Priority Management](#)

### Related topics

- [mdiag -p](#) (Priority Diagnostics)

## Priority Overview

Moab's prioritization mechanism allows component and subcomponent weights to be associated with many aspects of a job to enable fine-grained control over this aspect of scheduling. To allow this level of control, Moab uses a simple priority-weighting hierarchy where the contribution of each priority subcomponent is calculated as follows:

$$\langle \text{COMPONENT WEIGHT} \rangle * \langle \text{SUBCOMPONENT WEIGHT} \rangle * \langle \text{PRIORITY SUBCOMPONENT VALUE} \rangle$$

Each priority component contains one or more subcomponents as described in the section titled [Job Priority Factors on page 378](#). For example, the Resource component consists of Node, Processor, Memory, Swap, Disk, Walltime, and PE subcomponents. While there are numerous priority components and many more subcomponents, a site need only focus on and configure the subset of components related to their particular priority needs. In actual usage, few sites use more than a small fraction (usually 5 or fewer) of the available priority subcomponents. This results in fairly straightforward priority configurations and tuning. By mixing and matching priority weights, sites may generally obtain the desired job-start behavior. At any time, you can issue the [mdiag -p](#) command to determine the impact of the current priority-weight settings on idle jobs. Likewise, the command [showstats -f](#) can assist the administrator in evaluating priority effectiveness on historical system usage metrics such as queue time or expansion factor.

As mentioned above, a job's priority is the weighted sum of its activated subcomponents. By default, the value of all component and subcomponent weights is set to 1 and 0 respectively. The one exception is the "QUEUE TIME" subcomponent weight that is set to 1. This results in a total job priority equal to the period of time the job has been queued, causing Moab to act as a simple FIFO. Once the summed component weight is determined, this value is then bounded resulting in a priority ranging between 0

and `MAX_PRIO_VAL` which is currently defined as 1000000000 (one billion). In no case will a job obtain a priority in excess of `MAX_PRIO_VAL` through its priority subcomponent values.

 Negative priority jobs may be allowed if desired; see [ENABLENEGJOBPRIORITY](#) and [REJECTNEGPRIOJOBS](#) for more information.

Using the `mjobctl -p` command, site administrators may adjust the base calculated job priority by either assigning a relative priority adjustment or an absolute system priority. A relative priority adjustment causes the base priority to be increased or decreased by a specified value. Setting an absolute system priority, `SPRIO`, causes the job to receive a priority equal to `MAX_PRIO_VAL + SPRIO`, and thus guaranteed to be of higher value than any naturally occurring job priority.

#### Related topics

- [REJECTNEGPRIOJOBS](#) parameter

## Job Priority Factors

- [Credential \(CRED\) Component](#)
- [Fairshare \(FS\) Component](#)
- [Resource \(RES\) Component](#)
- [Service \(SERVICE\) Component](#)
- [Target Service \(TARG\) Component](#)
- [Usage \(USAGE\) Component](#)
- [Job Attribute \(ATTR\) Component](#)

Moab allows jobs to be prioritized based on a range of job related factors. These factors are broken down into a two-tier hierarchy of priority factors and subfactors, each of which can be independently assigned a weight. This approach provides the administrator with detailed yet straightforward control of the job selection process.

Each factor and subfactor can be configured with independent priority weight and priority [cap](#) values (described later). In addition, per credential and per QoS priority weight adjustments may be specified for a subset of the priority factors. For example, QoS credentials can adjust the queuetime subfactor weight and group credentials can adjust fairshare subfactor weight.

The following table highlights the factors and subfactors that make up a job's total priority.

Factor	SubFactor	Metric
<b><u>CRED</u></b> <b>(job credentials)</b>	USER	user-specific priority (See <a href="#">USERCFG</a> )
	GROUP	group-specific priority (See <a href="#">GROUPCFG</a> )
	ACCOUNT	account-specific priority (See <a href="#">ACCOUNTCFG</a> )
	QOS	QoS-specific priority (See <a href="#">QOSCFG</a> )
	CLASS	class/queue-specific priority (See <a href="#">CLASSCFG</a> )

Factor	SubFactor	Metric
<b>FS</b> <b>(fairshare usage)</b>	FSUSER	user-based historical usage (See <a href="#">Fairshare Overview</a> )
	FSGROUP	group-based historical usage (See <a href="#">Fairshare Overview</a> )
	FSACCOUNT	account-based historical usage (See <a href="#">Fairshare Overview</a> )
	FSQOS	QoS-based historical usage (See <a href="#">Fairshare Overview</a> )
	FSCCLASS	class/queue-based historical usage (See <a href="#">Fairshare Overview</a> )
	FSGUSER	imported global user-based historical usage (See <a href="#">ID Manager</a> and <a href="#">Fairshare Overview</a> )
	FSGGROUP	imported global group-based historical usage (See <a href="#">ID Manager</a> and <a href="#">Fairshare Overview</a> )
	FSGACCOUNT	imported global account-based historical usage (See <a href="#">ID Manager</a> and <a href="#">Fairshare Overview</a> )
	FSJPU	current active jobs associated with job user
	FSPPU	current number of processors allocated to active jobs associated with job user
	FSPSPU	current number of processor-seconds allocated to active jobs associated with job user
	WCACCURACY	user's current historical job wallclock accuracy calculated as total processor-seconds dedicated / total processor-seconds requested

 Factor values are in the range of 0.0 to 1.0.

Factor	SubFactor	Metric
<b><u>RES</u></b> <b>(requested job resources)</b>	NODE	number of nodes requested
	PROC	number of processors requested
	MEM	total real memory requested (in MB)
	SWAP	total virtual memory requested (in MB)
	DISK	total local disk requested (in MB)
	PS	total processor-seconds requested
	PE	total processor-equivalent requested
	WALLTIME	total walltime requested (in seconds)
<b><u>SERV</u></b> <b>(current service levels)</b>	<u>QUEUETIME</u>	time job has been queued (in minutes)
	<u>XFACTOR</u>	minimum job expansion factor
	<u>BYPASS</u>	number of times job has been bypassed by backfill
	<u>STARTCOUNT</u>	number of times job has been restarted
	<u>DEADLINE</u>	proximity to job deadline
	<u>SPVIOLATION</u>	Boolean indicating whether the active job violates a <u>soft</u> usage limit
	<u>USERPRIO</u>	user-specified job priority
<b><u>TARGET</u></b> <b>(target service levels)</b>	TARGETQUEUETIME	time until queuetime target is reached (exponential)
	TARGETXFACTOR	distance to target expansion factor (exponential)

Factor	SubFactor	Metric
<b>USAGE</b> (consumed resources -- active jobs only)	CONSUMED	processor-seconds dedicated to date
	REMAINING	processor-seconds outstanding
	PERCENT	percent of required walltime consumed
	EXECUTIONTIME	seconds since job started
<b>ATTR</b> (job attribute-based prioritization)	ATTRATTR	<a href="#">Attribute priority</a> if specified job attribute is set (attributes may be user-defined or one of <b>preemptor</b> , or <b>preemptee</b> ). Default is 0.
	ATTRSTATE	<a href="#">Attribute priority</a> if job is in specified state (see <a href="#">Job States</a> ). Default is 0.
	ATTRGRES	<a href="#">Attribute priority</a> if a <a href="#">generic resource</a> is requested. Default is 0.

**i** \*CAP parameters (**FSCAP**, for example) are available to limit the maximum absolute value of each priority component and subcomponent. If set to a positive value, a priority cap will bound priority component values in both the positive and negative directions.

**i** All \*CAP and \*WEIGHT parameters are specified as positive or negative integers. Non-integer values are not supported.

## Credential (CRED) Component

The credential component allows a site to prioritize jobs based on political issues such as the relative importance of certain groups or accounts. This allows direct political priorities to be applied to jobs.

The priority calculation for the credential component is as follows:

$$\text{Priority} += \text{CREDWEIGHT} * ( \text{USERWEIGHT} * \text{Job.User.Priority} + \text{GROUPWEIGHT} * \text{Job.Group.Priority} + \text{ACCOUNTWEIGHT} * \text{Job.Account.Priority} + \text{QOSWEIGHT} * \text{Job.Qos.Priority} + \text{CLASSWEIGHT} * \text{Job.Class.Priority} )$$

All user, group, account, QoS, and class weights are specified by setting the **PRIORITY** attribute of using the respective \*CFG parameter (namely, **USERCFG**, **GROUPCFG**, **ACCOUNTCFG**, **QOSCFG**, and **CLASSCFG**).

For example, to set user and group priorities, you might use the following:

```

CREDWEIGHT      1
USERWEIGHT      1
GROUPWEIGHT     1
USERCFG[john]   PRIORITY=2000
USERCFG[paul]   PRIORITY=-1000
GROUPCFG[staff] PRIORITY=10000

```

**i** Class (or queue) priority may also be specified via the resource manager where supported (as in PBS queue priorities). However, if Moab class priority values are also specified, the resource manager priority values will be overwritten.

All priorities may be positive or negative.

## Fairshare (FS) Component

Fairshare components allow a site to favor jobs based on short-term historical usage. The [Fairshare Overview](#) describes the configuration and use of fairshare in detail.

The fairshare factor is used to adjust a job's priority based on current and historical percentage system utilization of the job's user, group, account, class, or QoS. This allows sites to steer workload toward a particular usage mix across user, group, account, class, and QoS dimensions.

The fairshare priority factor calculation is as follows:

```

Priority += FSWEIGHT * MIN(FSCAP, (
  FSUSERWEIGHT * DeltaUserFSUsage +
  FSGROUPWEIGHT * DeltaGroupFSUsage +
  FSACCOUNTWEIGHT * DeltaAccountFSUsage +
  FSQOSWEIGHT * DeltaQOSFSUsage +
  FSCLASSWEIGHT * DeltaClassFSUsage +
  FSJPUWEIGHT * ActiveUserJobs +
  FSPPUWEIGHT * ActiveUserProcs +
  FSPSPUWEIGHT * ActiveUserPS +
  WCACCURACYWEIGHT * UserWCAccuracy ))

```

All **\*WEIGHT** parameters just listed are specified on a per partition basis in the `moab.cfg` file. The `Delta*Usage` components represent the difference in actual fairshare usage from the corresponding fairshare usage target. Actual fairshare usage is determined based on historical usage over the time frame specified in the fairshare configuration. The target usage can be a target, floor, or ceiling value as specified in the fairshare configuration file. See the [Fairshare Overview](#) for further information on configuring and tuning fairshare. Additional insight may be available in the [fairshare usage example](#). The `ActiveUser*` components represent current usage by the job's user credential.

How violated ceilings and floors affect fairshare-based priority

Moab determines `FSUsageWeight` in the previous section. In order to account for violated ceilings and floors, Moab multiplies that number by the `FSUsagePriority` as demonstrated in the following formula:

$$FSPriority = FSUsagePriority * FSUsageWeight$$

When a ceiling or floor is violated, `FSUsagePriority = 0`, so `FSPriority = 0`. This means the job will gain no priority because of fairshare. If fairshare is the only component of priority, then violation

takes the priority to 0. For more information, see [Priority-Based Fairshare on page 439](#) and [Fairshare Targets on page 436](#).

## Resource (RES) Component

Weighting jobs by the amount of resources requested allows a site to favor particular types of jobs. Such prioritization may allow a site to better meet site mission objectives, improve fairness, or even improve overall system utilization.

Resource based prioritization is valuable when you want to favor jobs based on the resources requested. This is good in three main scenarios: (1) when you need to favor large resource jobs because it's part of your site's mission statement, (2) when you want to level the response time distribution across large and small jobs (small jobs are more easily backfilled and thus generally have better turnaround time), and (3) when you want to improve system utilization. While this may be surprising, system utilization actually increases as large resource jobs are pushed to the front of the queue. This keeps the smaller jobs in the back where they can be selected for backfill and thus increase overall system utilization. The situation is like the story about filling a cup with golf balls and sand. If you put the sand in first, it gets in the way and you are unable to put in as many golf balls. However, if you put in the golf balls first, the sand can easily be poured in around them completely filling the cup.

The calculation for determining the total resource priority factor is as follows:

$$\text{Priority} += \text{RESWEIGHT} * \text{MIN}(\text{RESCAP}, ($$

$$\text{NODEWEIGHT} * \text{TotalNodesRequested} +$$

$$\text{PROCWEIGHT} * \text{TotalProcessorsRequested} +$$

$$\text{MEMWEIGHT} * \text{TotalMemoryRequested} +$$

$$\text{SWAPWEIGHT} * \text{TotalSwapRequested} +$$

$$\text{DISKWEIGHT} * \text{TotalDiskRequested} +$$

$$\text{WALLTIMEWEIGHT} * \text{TotalWalltimeRequested} +$$

$$\text{PEWEIGHT} * \text{TotalPERequested}))$$

The sum of all weighted resources components is then multiplied by the **RESWEIGHT** parameter and capped by the **RESCAP** parameter. Memory, Swap, and Disk are all measured in megabytes (MB). The final resource component, PE, represents [Processor Equivalents](#). This component can be viewed as a processor-weighted maximum *percentage of total resources* factor.

For example, if a job requested 25% of the processors and 50% of the total memory on a 128-processor system, it would have a PE value of  $\text{MAX}(25,50) * 128$ , or 64. The concept of PEs is a highly effective metric in shared resource systems.



Ideal values for requested job processor count and walltime can be specified using [PRIORITYTARGETPROCCOUNT](#) and [PRIORITYTARGETDURATION](#).

## Service (SERVICE) Component

The Service component specifies which service metrics are of greatest value to the site. Favoring one service subcomponent over another generally improves that service metric.

The priority calculation for the service priority factor is as follows:

$$\text{Priority} += \text{SERVICEWEIGHT} * ($$

$$\text{QUEUETIMEWEIGHT} * \langle \text{QUEUETIME} \rangle +$$

[XFACTORWEIGHT](#) \* <XFACTOR> +  
[BYPASSWEIGHT](#) \* <BYPASSCOUNT> +  
[STARTCOUNTWEIGHT](#) \* <STARTCOUNT> +  
[DEADLINEWEIGHT](#) \* <DEADLINE> +  
[SPVIOLATIONWEIGHT](#) \* <SPBOOLEAN> +  
[USERPRIOWEIGHT](#) \* <USERPRIO> )

## QueueTime (QUEUE TIME) Subcomponent

In the priority calculation, a job's queue time is a duration measured in minutes. Using this subcomponent tends to prioritize jobs in a FIFO order. Favoring queue time improves queue time based fairness metrics and is probably the most widely used single job priority metric. In fact, under the initial default configuration, this is the only priority subcomponent enabled within Moab. It is important to note that within Moab, a job's queue time is not necessarily the amount of time since the job was submitted. The parameter [JOBPRIOACCRUALPOLICY](#) allows a site to select how a job will accrue queue time based on meeting various [throttling policies](#). Regardless of the policy used to determine a job's queue time, this effective queue time is used in the calculation of the [QUEUE TIME](#), [XFACTOR](#), [TARGET QUEUE TIME](#), and [TARGET XFACTOR](#) priority subcomponent values.

The need for a distinct effective queue time is necessitated by the fact that many sites have users who like to work the system, whatever system it happens to be. A common practice at some long existing sites is for some users to submit a large number of jobs and then place them on hold. These jobs remain with a hold in place for an extended period of time and when the user is ready to run a job, the needed executable and data files are linked into place and the hold released on one of these pre-submitted jobs. The extended hold time guarantees that this job is now the highest priority job and will be the next to run. The use of the [JOBPRIOACCRUALPOLICY](#) parameter can prevent this practice and prevent "queue stuffers" from doing similar things on a shorter time scale. These "queue stuffer" users submit hundreds of jobs at once to swamp the machine and consume use of the available compute resources. This parameter prevents the user from gaining any advantage from stuffing the queue by not allowing these jobs to accumulate any queue time based priority until they meet certain idle and active Moab fairness policies (such as max job per user and max idle job per user).

As a final note, you can adjust the [QUEUE TIME WEIGHT](#) parameter on a per QoS basis using the [QOSCFG](#) parameter and the [QTWEIGHT](#) attribute. For example, the line [QOSCFG\[\*special\*\] QTWEIGHT=5000](#) causes jobs using the QoS *special* to have their queue time subcomponent weight increased by 5000.

## Expansion Factor (XFACTOR) Subcomponent

The expansion factor subcomponent has an effect similar to the queue time factor but favors shorter jobs based on their requested wallclock run time. In its traditional form, the expansion factor (XFactor) metric is calculated as follows:

$$\text{XFACTOR} = 1 + \frac{\langle \text{QUEUE TIME} \rangle}{\langle \text{EXECUTION TIME} \rangle}$$

However, a couple of aspects of this calculation make its use more difficult. First, the length of time the job will actually run—[EXECUTION TIME](#)—is not actually known until the job completes. All that is known is how much time the job requests. Secondly, as described in the [Queue Time Subcomponent](#) section, Moab does not necessarily use the raw time since job submission to determine [QUEUE TIME](#) to prevent various scheduler abuses. Consequently, Moab uses the following modified equation:

$$\text{XFACTOR} = 1 + \frac{\langle \text{EFF QUEUE TIME} \rangle}{\langle \text{WALLCLOCK LIMIT} \rangle}$$

In the equation Moab uses, <EFFQUEUEETIME> is the effective queue time subject to the [JOBPRIOACCRUALPOLICY](#) parameter and <WALLCLOCKLIMIT> is the user—or system—specified job wallclock limit.

Using this equation, it can be seen that short running jobs will have an XFactor that will grow much faster over time than the xfactor associated with long running jobs. The following table demonstrates this favoring of short running jobs:

Job Queue Time	1 hour	2 hours	4 hours	8 hours	16 hours
<b>XFactor for 1 hour job</b>	$1 + (1 / 1) = 2.00$	$1 + (2 / 1) = 3.00$	$1 + (4 / 1) = 5.00$	$1 + (8 / 1) = 9.00$	$1 + (16 / 1) = 17.0$
<b>XFactor for 4 hour job</b>	$1 + (1 / 4) = 1.25$	$1 + (2 / 4) = 1.50$	$1 + (4 / 4) = 2.00$	$1 + (8 / 4) = 3.00$	$1 + (16 / 4) = 5.0$

Since XFactor is calculated as a ratio of two values, it is possible for this subcomponent to be almost arbitrarily large, potentially swamping the value of other priority subcomponents. This can be addressed either by using the subcomponent cap [XFACTORCAP](#), or by using the [XFMINWCLIMIT](#) parameter. If the latter is used, the calculation for the XFactor subcomponent value becomes:

$$XFACTOR = 1 + \text{<EFFQUEUEETIME>} / \text{MAX(}<XFMINWCLIMIT>, \text{<WALLCLOCKLIMIT>)}$$

Using the [XFMINWCLIMIT](#) parameter allows a site to prevent very short jobs from causing the XFactor subcomponent to grow inordinately.

Some sites consider XFactor to be a more fair scheduling performance metric than queue time. At these sites, job XFactor is given far more weight than job queue time when calculating job priority and job XFactor distribution consequently tends to be fairly level across a wide range of job durations. (That is, a flat XFactor distribution of 1.0 would result in a one-minute job being queued on average one minute, while a 24-hour job would be queued an average of 24 hours.)

Like queue time, the effective XFactor subcomponent weight is the sum of two weights, the [XFACTORWEIGHT](#) parameter and the QoS-specific XFWEIGHT setting. For example, the line [QOSCFG \[special\] XFWEIGHT=5000](#) causes jobs using the QoS *special* to increase their expansion factor subcomponent weight by 5000.

### Bypass (BYPASS) Subcomponent

The bypass factor is based on the bypass count of a job where the bypass count is increased by one every time the job is bypassed by a lower priority job via backfill. Backfill starvation has never been reported, but if encountered, use the BYPASS subcomponent.

### StartCount (STARTCOUNT) Subcomponent

Apply the startcount factor to sites with trouble starting or completing due to policies or failures. The primary causes of an idle job having a startcount greater than zero are resource manager level job start failure, administrator based requeue, or requeue based preemption.

## Deadline (DEADLINE) Subcomponent

The deadline factor allows sites to take into consideration the proximity of a job to its [DEADLINE](#). As a jobs moves closer to its deadline its priority increases linearly. This is an alternative to the strict deadline discussed in [QOS SERVICE](#).

## Soft Policy Violation (SPVIOLATION) Subcomponent

The soft policy violation factor allows sites to favor jobs which do not violate their associated [soft resource limit policies](#).

## User Priority (USERPRIO) Subcomponent

The user priority subcomponent allows sites to consider end-user specified job priority in making the overall job priority calculation. Under Moab, end-user specified priorities may only be negative and are bounded in the range 0 to -1024. See [Manual Priority Usage](#) and [Enabling End-user Priorities](#) for more information.



User priorities can be positive, ranging from -1024 to 1023, if [ENABLEPOSUSERPRIORITY](#) TRUE is specified in `moab.cfg`.

## Target Service (TARG) Component

The target factor component of priority takes into account job scheduling performance targets. Currently, this is limited to target expansion factor and target queue time. Unlike the expansion factor and queue time factors described earlier which increase gradually over time, the target factor component is designed to grow exponentially as the target metric is approached. This behavior causes the scheduler to do essentially all in its power to make certain the scheduling targets are met.

The priority calculation for the target factor is as follows:

$$\text{Priority} += \frac{\text{TARGETWEIGHT}}{\text{TARGETQUEUE TIMEWEIGHT}} * \text{QueueTimeComponent} + \frac{\text{TARGETXFACTORWEIGHT}}{\text{TARGETXFACTORWEIGHT}} * \text{XFactorComponent}$$

The queue time and expansion factor target are specified on a per QoS basis using the **XFTARGET** and **QTTARGET** attributes with the [QOSCFG](#) parameter. The QueueTime and XFactor component calculations are designed to produce small values until the target value begins to approach, at which point these components grow very rapidly. If the target is missed, this component remains high and continues to grow, but it does not grow exponentially.

## Usage (USAGE) Component

The Usage component applies to active jobs only. The priority calculation for the usage priority factor is as follows:

$$\text{Priority} += \frac{\text{USAGEWEIGHT}}{\text{USAGECONSUMEDWEIGHT}} * \text{ProcSecondsConsumed} + \frac{\text{USAGEHUNGERWEIGHT}}{\text{USAGEHUNGERWEIGHT}} * \text{ProcNeededToBalanceDynamicJob} + \frac{\text{USAGEREMAININGWEIGHT}}{\text{USAGEREMAININGWEIGHT}} * \text{ProcSecRemaining} +$$

[USAGEEXECUTIONTIMEWEIGHT](#) \* SecondsSinceStart +  
[USAGEPERCENTWEIGHT](#) \* WalltimePercent )

## Job Attribute (ATTR) Component

The Attribute component allows the incorporation of job attributes into a job's priority. The most common usage for this capability is to do one of the following:

- adjust priority based on a job's state (favor suspended jobs)
- adjust priority based on a job's requested node features (favor jobs that request attribute **pvfs**)
- adjust priority based on internal job attributes (disfavor **backfill** or **preemptee** jobs)
- adjust priority based on a job's requested licenses, network consumption, or generic resource requirements

To use job attribute based prioritization, the [JOBPRIOF](#) parameter must be specified to set corresponding attribute priorities. To favor jobs based on node feature requirements, the parameter [NODETOJOBATTRMAP](#) must be set to map node feature requests to job attributes.

The priority calculation for the attribute priority factor is as follows:

Priority += [ATTRWEIGHT](#) \* (  
[ATTRATTRWEIGHT](#) \* <ATTRPRIORITY> +  
[ATTRSTATEWEIGHT](#) \* <STATEPRIORITY> +  
[ATTRGRESWEIGHT](#) \* <GRESPRIORITY>  
[JOBIDWEIGHT](#) \* <JOBID> +  
[JOBNAMEWEIGHT](#) \* <JOBNAME\_INTEGER> )

Example 3-88:

```
ATTRWEIGHT      100
ATTRATTRWEIGHT  1
ATTRSTATEWEIGHT 1
ATTRGRESWEIGHT  5
# favor suspended jobs
# disfavor preemptible jobs
# favor jobs requesting 'matlab'

JOBPRIOF STATE[Running]=100 STATE[Suspended]=1000 ATTR[PREEMPTEE]=-200 ATTR[gpfs]
=30 GRES[matlab]=400
# map node features to job features

NODETOJOBATTRMAP gpfs,pvfs
...
```

### Related topics

- [Node Allocation Priority](#)
- [Per Credential Priority Weight Offsets](#)
- [Managing Consumable Generic Resources](#)

## Fairshare Job Priority Example

Consider the following information associated with calculating the fairshare factor for job X.

Job X

User A  
 Group B  
 Account C  
 QOS D  
 Class E

User A

Fairshare Target: 50.0  
 Current Fairshare Usage: 45.0

Group B

Fairshare Target: [NONE]  
 Current Fairshare Usage: 65.0

Account C

Fairshare Target: 25.0  
 Current Fairshare Usage: 35.0

QOS D

Fairshare Target: 10.0+  
 Current Fairshare Usage: 25.0

Class E

Fairshare Target: [NONE]  
 Current Fairshare Usage: 20.0

Priority Weights:

FSWEIGHT 100  
 FSUSERWEIGHT 10  
 FSGROUPWEIGHT 20  
 FSACCOUNTWEIGHT 30  
 FSQOSWEIGHT 40  
 FSCLASSWEIGHT 0

In this example, the Fairshare component calculation would be as follows:

$$\text{Priority} += 100 * ( 10 * 5 + 20 * 0 + 30 * (-10) + 40 * 0 + 0 * 0 )$$

User A is 5% below his target so fairshare increases the total fairshare factor accordingly. Group B has no target so group fairshare usage is ignored. Account C is 10% above its fairshare usage target so this component decreases the job's total fairshare factor. QOS D is 15% over its target but the '+' in the target specification indicates that this is a 'floor' target, only influencing priority when fairshare usage drops below the target value. Thus, the QOS D fairshare usage delta does not influence the fairshare factor.

Fairshare is a great mechanism for influencing job turnaround time via priority to favor a particular distribution of jobs. However, it is important to realize that fairshare can only favor a particular

distribution of jobs, it cannot force it. If user X has a fairshare target of 50% of the machine but does not submit enough jobs, no amount of priority favoring will get user X's usage up to 50%.

See the [Fairshare Overview](#) for more information.

## Common Priority Usage

- [Credential Priority Factors](#)
- [Service Level Priority Factors](#)
- [Priority Factor Caps](#)
- [User Selectable Prioritization](#)

Site administrators vary widely in their preferred manner of prioritizing jobs. Moab's scheduling hierarchy allows sites to meet job control needs without requiring adjustments to dozens of parameters. Some choose to use numerous subcomponents, others a few, and still others are content with the default FIFO behavior. Any subcomponent that is not of interest may be safely ignored.

## Credential Priority Factors

To help clarify the use of priority weights, a brief example may help. Suppose a site wished to maintain the FIFO behavior but also incorporate some credential based prioritization to favor a special user. Particularly, the site would like the user *john* to receive a higher initial priority than all other users. Configuring this behavior requires two steps. First, the user credential subcomponent must be enabled and second, *john* must have his relative priority specified. Take a look at the sample `moab.cfg` file:

```
USERWEIGHT      1
USERCFG[john]   PRIORITY=300
```

**i** The "USER" priority subcomponent was enabled by setting the [USERWEIGHT](#) parameter. In fact, the parameters used to specify the weights of all components and subcomponents follow this same "\*WEIGHT" naming convention (as in [RESWEIGHT](#) and [TARGETQUEUEWEIGHT](#)).

The second part of the example involves specifying the actual user priority for the user *john*. This is accomplished using the [USERCFG](#) parameter. Why was the priority 300 selected and not some other value? Is this value arbitrary? As in any priority system, actual priority values are meaningless, only relative values are important. In this case, we are required to balance user priorities with the default queue time based priorities. Since queue time priority is measured in minutes queued, the user priority of 300 places a job by user *john* on par with a job submitted 5 minutes earlier by another user.

Is this what the site wants? Maybe, maybe not. At the onset, most sites are uncertain what they want in prioritization. Often, an estimate initiates prioritization and adjustments occur over time. Cluster resources evolve, the workload evolves, and even site policies evolve, resulting in changing priority needs over time. Anecdotal evidence indicates that most sites establish a relatively stable priority policy within a few iterations and make only occasional adjustments to priority weights from that point.

## Service Level Priority Factors

In another example, suppose a site administrator wants to do the following:

- favor jobs in the low, medium, and high QoSs so they will run in QoS order
- balance job expansion factor
- use job queue time to prevent jobs from starving

Under such conditions, the sample `moab.cfg` file might appear as follows:

```

QOSWEIGHT          1
XFACTORWEIGHT      1
QUEUEUETIMEWEIGHT 10
TARGETQUEUEUETIMEWEIGHT 1
QOSCFG [low]        PRIORITY=1000
QOSCFG [medium]     PRIORITY=10000
QOSCFG [high]       PRIORITY=100000
QOSCFG [DEFAULT]   QTTARGET=4:00:00

```

This example is a bit more complicated but is more typical of the needs of many sites. The desired QoS weightings are established by enabling the QoS subfactor using the `QOSWEIGHT` parameter while the various QoS priorities are specified using `QOSCFG`. `XFACTORWEIGHT` is then set as this subcomponent tends to establish a balanced distribution of expansion factors across all jobs. Next, the queue time component is used to gradually raise the priority of all jobs based on the length of time they have been queued. Note that in this case, `QUEUEUETIMEWEIGHT` was explicitly set to 10, overriding its default value of 1. Finally, the `TARGETQUEUEUETIMEWEIGHT` parameter is used in conjunction with the `USERCFG` line to specify a queue time target of 4 hours.

## Priority Factor Caps

Assume now that the site administrator is content with this priority mix but has a problem with users submitting large numbers of very short jobs. Very short jobs would tend to have rapidly growing XFactor values and would consequently quickly jump to the head of the queue. In this case, a factor cap would be appropriate. Such caps allow a site to limit the contribution of a job's priority factor to be within a defined range. This prevents certain priority factors from swamping others. Caps can be applied to either priority components or subcomponents and are specified using the `<COMPONENTNAME>CAP` parameter (such as `QUEUEUETIMECAP`, `RESCAP`, and `SERVCAP`). Note that both component and subcomponent caps apply to the pre-weighted value, as in the following equation:

```

Priority =
  C1WEIGHT * MIN(C1CAP, SUM(
    S11WEIGHT * MIN(S11CAP, S11S) +
    S12WEIGHT * MIN(S12CAP, S12S) +
    ...)) +
  C2WEIGHT * MIN(C2CAP, SUM(
    S21WEIGHT * MIN(S21CAP, S21S) +
    S22WEIGHT * MIN(S22CAP, S22S) +
    ...)) +
  ...

```

*Example 3-89: Priority cap*

```

QOSWEIGHT      1
QOSCAP         10000
XFACTORWEIGHT  1
XFACTORCAP     1000
QUEUE TIMEWEIGHT 10
QUEUE TIMECAP  1000

```

## User Selectable Prioritization

Moab allows users to specify a job priority to jobs they own or manage. This priority may be set at job submission time or it may be dynamically modified (using [setspri](#) or [mjobctl](#)) after submitting the job. For fairness reasons, users may only apply a negative priority to their job and thus slide it further back in the queue. This enables users to allow their more important jobs to run before their less important ones without gaining unfair advantage over other users.

**i** User priorities can be positive if [ENABLEPOSUSERPRIORITY](#) TRUE is specified in `moab.cfg`. In order to set **ENABLEPOSUSERPRIORITY**, you must change the [USERPRIOWEIGHT](#) from its default value of 0. For example:

```
USERPRIOWEIGHT 100
```

```
> setspri -r 100 332411
successfully modified job priority
```

**i** Specifying a user priority at job submission time is resource manager specific. See the associated resource manager documentation for more information.

## User Selectable Priority w/QoS

Using the [QoS](#) facility, organizations can set up an environment in which users can more freely select the desired priority of a given job. Organizations may enable access to a number of QoSs each with its own charging rate, priority, and target service levels. Users can then assign job importance by selecting the appropriate QoS. If desired, this can allow a user to jump ahead of other users in the queue if they are willing to pay the associated costs.

### Related topics

- [User Selectable Priority](#)

## Prioritization Strategies

Each component or subcomponent may be used to accomplish different objectives. **WALLTIME** can be used to favor (or disfavor) jobs based on their duration. Likewise, **ACCOUNT** can be used to favor jobs associated with a particular project while **QUEUE TIME** can be used to favor those jobs waiting the longest.

- Queue Time
- Expansion Factor
- Resource
- Fairshare
- Credential
- Target Metrics

Each priority factor group may contain one or more subfactors. For example, the Resource factor consists of Node, Processor, Memory, Swap, Disk, and PE components. From the table in [Job Priority Factors](#) section, it is apparent that the prioritization problem is fairly complex since every site needs to prioritize a bit differently. When calculating a priority, the various priority factors are summed and then bounded between 0 and MAX\_PRIO\_VAL, which is currently defined as 100000000 (one billion).

The [mdiag -p](#) command assists with visualizing the priority distribution resulting from the current job priority configuration. Also, the [showstats -f](#) command helps indicate the impact of the current priority settings on scheduler service distributions.

## Manual Job Priority Adjustment

Batch administrator's regularly find a need to adjust the calculated priority of a job to meet current needs. Current needs often are broken into two categories:

1. The need to run an administrator test job as soon as possible.
2. The need to pacify a disserved user.

You can use the [setspri](#) command to handle these issues in one of two ways; this command allows the specification of either a relative priority adjustment or the specification of an absolute priority. Using absolute priority specification, administrators can set a job priority guaranteed to be higher than any calculated value. Where Moab-calculated job priorities are in the range of 0 to 1 billion, system administrator assigned absolute priorities start at 1 billion and go up. Issuing the `setspri <PRIO> <JOBID>` command, for example, assigns a priority of 1 billion + <PRIO> to the job. Thus, `setspri 5 job.1294` sets the priority of "job.1294" to 1000000005.

For more information, see [Common Priority Usage - End-user Adjustment](#).

## Node Allocation Policies

While job prioritization allows a site to determine which job to run, node allocation policies allow a site to specify how available resources should be allocated to each job. The algorithm used is specified by the parameter [NODEALLOCATIONPOLICY](#). There are multiple node allocation policies to choose from allowing selection based on reservation constraints, node configuration, resource usage, preferred other factors. You can specify these policies with a system-wide default value, on a per-partition basis, or on a per-job basis. Please note that **LASTAVAILABLE** is the default policy.

Available algorithms are described in detail in the following sections and include [FIRSTAVAILABLE](#), [LASTAVAILABLE](#), [PRIORITY](#), [CPULOAD](#), [MINRESOURCE](#), [CONTIGUOUS](#), [MAXBALANCE](#), [PLUGIN](#).

- [Node Allocation Overview](#)
  - [Heterogeneous Resources](#)
  - [Shared Nodes](#)
  - [Reservations or Service Guarantees](#)
  - [Non-flat Network](#)
- [Node selection factors on page 398](#)
- [Resource-Based Algorithms](#)
  - [CPULOAD](#)
  - [FIRSTAVAILABLE](#)
  - [LASTAVAILABLE](#)
  - [PRIORITY](#)
  - [MINRESOURCE](#)
  - [CONTIGUOUS](#)
  - [MAXBALANCE](#)
- [User-Defined Algorithms](#)
  - [PLUGIN](#)
- [Specifying Per Job Resource Preferences](#)
  - [Specifying Resource Preferences](#)
  - [Selecting Preferred Resources](#)

## Node Allocation Overview

Node allocation is the process of selecting the best resources to allocate to a job from a list of available resources. Making this decision intelligently is important in an environment that possesses one or more of the following attributes:

- heterogeneous resources (resources which vary from node to node in terms of quantity or quality)
- shared nodes (nodes may be utilized by more than one job)
- reservations or service guarantees
- non-flat network (a network in which a perceptible performance degradation may potentially exist depending on workload placement)

## Heterogeneous Resources

Moab analyzes job processing requirements and assigns resources to maximize hardware utility.

For example, suppose two nodes are available in a system, A and B. Node A has 768 MB of RAM and node B has 512 MB. The next two jobs in the queue are X and Y. Job X requests 256 MB and job Y requests 640

MB. Job X is next in the queue and can fit on either node, but Moab recognizes that job Y (640 MB) can only fit on node A (768 MB). Instead of putting job X on node A and blocking job Y, Moab can put job X on node B and job Y on node A.

## Shared Nodes

### *Symmetric Multiprocessing (SMP)*

When sharing SMP-based compute resources amongst tasks from more than one job, resource contention and fragmentation issues arise. In SMP environments, the general goal is to deliver maximum system utilization for a combination of compute-intensive and memory-intensive jobs while preventing overcommitment of resources.

By default, most current systems do not do a good job of logically partitioning the resources (such as CPU, memory, and network bandwidth) available on a given node. Consequently contention often arises between tasks of independent jobs on the node. This can result in a slowdown for all jobs involved, which can have significant ramifications if large-way parallel jobs are involved. Virtualization, CPU sets, and other techniques are maturing quickly as methods to provide logical partitioning within shared resources.

On large-way SMP systems (> 32 processors/node), job packing can result in intra-node fragmentation. For example, take two nodes, A and B, each with 64 processors. Assume they are currently loaded with various jobs and A has 24 and B has 12 processors free. Two jobs are submitted; job X requests 10 processors and job Y requests 20 processors. Job X can start on either node but starting it on node A prevents job Y from running. An algorithm to handle intra-node fragmentation is straightforward for a single resource case, but the algorithm becomes more involved when jobs request a combination of processors, memory, and local disk. These workload factors should be considered when selecting a site's node allocation policy as well as identifying appropriate policies for handling resource utilization limit violations.

### *Interactive Nodes*

In many cases, sites are interested in allowing multiple users to simultaneously use one or more nodes for interactive purposes. Workload is commonly not compute intensive consisting of intermittent tasks including coding, compiling, and testing. Because these jobs are highly variant in terms of resource usage over time, sites are able to pack a larger number of these jobs onto the same node. Consequently, a common practice is to restrict job scheduling based on utilized, rather than dedicated resources.

### *Interactive Node Example*

The example configuration files that follow show one method by which node sharing can be accomplished within a [TORQUE](#) + Moab environment. This example is based on a hypothetical cluster composed of 4 nodes each with 4 cores. For the compute nodes, job tasks are limited to actual cores preventing overcommitment of resources. For the interactive nodes, up to 32 job tasks are allowed, but the node also stops allowing additional tasks if either memory is fully utilized or if the CPU load exceeds 4.0. Thus, Moab continues packing the interactive nodes with jobs until carrying capacity is reached.

*Example 3-90: /opt/moab/etc/moab.cfg*

```
# constrain interactive jobs to interactive nodes
# constrain interactive jobs to 900 proc-seconds
CLASSCFG[interactive]  HOSTLIST=interactive01,interactive02
CLASSCFG[interactive]  MAX.CPUTIME=900
RESOURCELIMITPOLICY    CPUTIME:ALWAYS:CANCEL
# base interactive node allocation on load and jobs
NODEALLOCATIONPOLICY    PRIORITY
NODECFG[interactive01] PRIORITYF='-20*LOAD - JOBCOUNT'
NODECFG[interactive02] PRIORITYF='-20*LOAD - JOBCOUNT'
```

*Example 3-91: /var/spool/torque/server\_priv/nodes*

```
interactive01 np=32
interactive02 np=32
compute01    np=4
compute02    np=4
```

*Example 3-92: /var/spool/torque/mom\_priv/config on "interactive01"*

```
# interactive01
$max_load 4.0
```

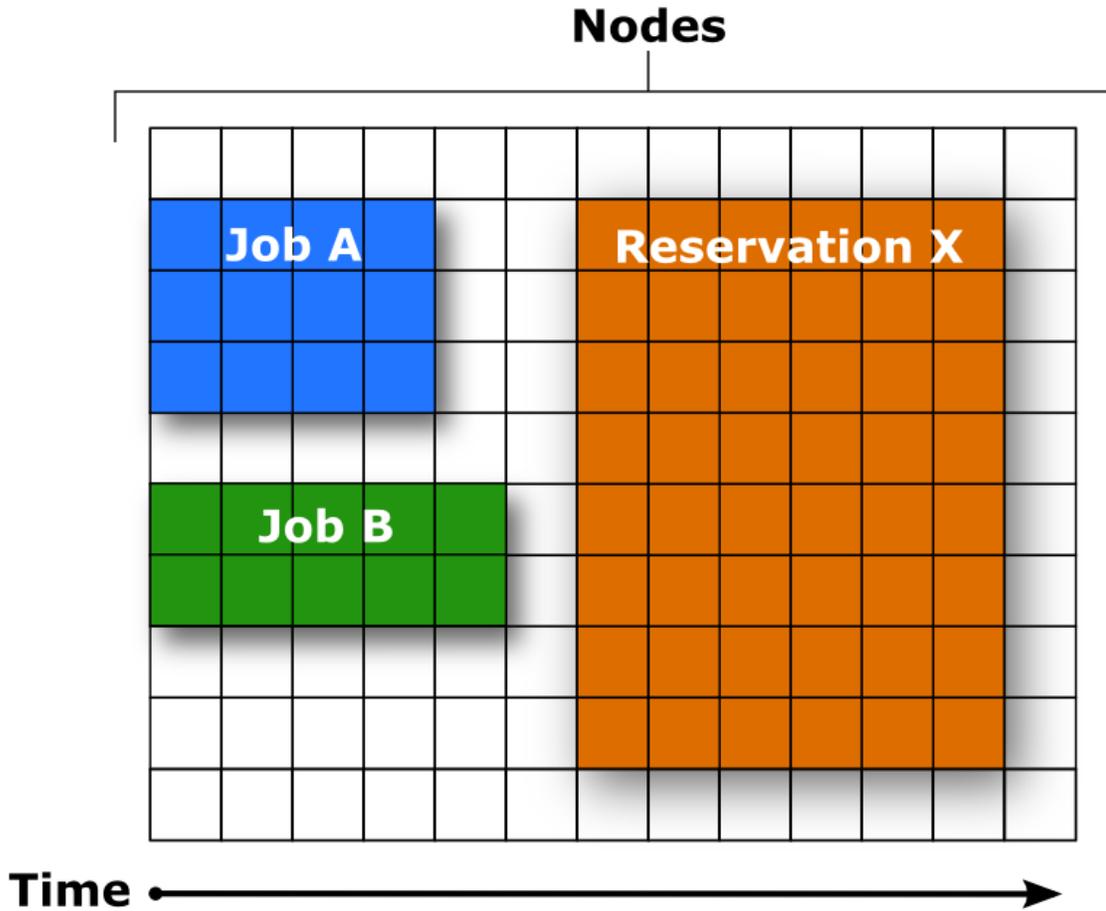
*Example 3-93: /var/spool/torque/mom\_priv/config on "interactive02"*

```
# interactive02
$max_load 4.0
```

**Reservations or Service Guarantees**

A reservation-based system adds the time dimension into the node allocation decision. With reservations, node resources must be viewed in a type of two dimension node-time space. Allocating nodes to jobs fragments this node-time space and makes it more difficult to schedule jobs in the remaining, more constrained node-time slots. Allocation decisions should be made in such a way as to minimize this fragmentation and maximize the scheduler's ability to continue to start jobs in existing slots. The following figure shows that job A and job B are running. A reservation, X, is created some time in the future. Assume that job A is 2 hours long and job B is 3 hours long. Again, two new single-processor jobs are submitted, C and D; job C requires 3 hours of compute time while job D requires 5 hours. Either job will just fit in the free space located above job A or in the free space located below job B. If job C is placed above job A, job D, requiring 5 hours of time will be prevented from running by the presence of reservation X. However, if job C is placed below job B, job D can still start immediately above job A.

Image 3-2: Job A, Job B, and Reservation X scheduled on nodes



The preceding example demonstrates the importance of time based reservation information in making node allocation decisions, both at the time of starting jobs and at the time of creating reservations. The impact of time based issues grows significantly with the number of reservations in place on a given system. The **LASTAVAILABLE** algorithm works on this premise, locating resources that have the smallest space between the end of a job under consideration and the start of a future reservation.

### Non-flat Network

On systems where network connections do not resemble a flat all-to-all topology, task placement may impact performance of communication intensive parallel jobs. If latencies and network bandwidth between any two nodes vary significantly, the node allocation algorithm should attempt to pack tasks of a given job as close to each other as possible to minimize impact of bandwidth and latency differences.

## Node selection factors

While the node allocation policy determines which nodes a job will use, other factors narrow the options before the policy makes the final decision. The following process demonstrates how Moab executes its node allocation process and how other policies affect the decision:

1. Moab eliminates nodes that do not meet the hard resource requirements set by the job.
2. Moab gathers affinity information, first from workload proximity rules and then from reservation affinity rules (See [Affinity on page 485](#) for more information.). Reservation affinity rules trump workload proximity rules.
3. Moab allocates nodes using the allocation policy.
  - If more than enough nodes with Required affinity exist, only they are passed down for the final sort by the node allocation policy.
  - If the number of nodes with Required affinity matches the number of nodes requested exactly, then the node allocation policy is skipped entirely and all of those nodes are assigned to the job.
  - If too few nodes have Required affinity, all of them are assigned to the job, then the node allocation policy is applied to the remaining eligible nodes (after Required, Moab will use Positive, then Neutral, then Negative.).

## Resource-Based Algorithms

Moab contains a number of allocation algorithms that address some of the needs described earlier. You can also create allocation algorithms and interface them with the Moab scheduling system. Each of these policies has a name and descriptive alias. They can be configured using either one, but Moab will only report their names.

**i** If [ENABLEHIGHTHROUGHPUT on page 950](#) is *TRUE*, you must set [NODEALLOCATIONPOLICY on page 999](#) to *FIRSTAVAILABLE*.

The current suite of algorithms is described in what follows:

Allocation algorithm name	Alias	Description
<b>CPULOAD</b>	ProcessorLoad	Nodes are selected that have the maximum amount of available, unused CPU power (<#of CPU's> - <CPU load>). CPULOAD is a good algorithm for timesharing node systems and applies to jobs starting immediately. For the purpose of future reservations, the MINRESOURCE algorithm is used.
<b>FIRSTAVAILABLE</b>	InReportedOrder	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, and very fast algorithm.

Allocation algorithm name	Alias	Description
<b>LASTAVAILABLE</b>	InReserveReportedOrder	Nodes are allocated in descending order that they are presented by the resource manager, or the reverse of FIRSTAVAILABLE.

Allocation algorithm name	Alias	Description
<b>PRIORITY</b>	CustomPriority	<p>Allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity. Using node allocation priority, you can specify the following priority components:</p> <ul style="list-style-type: none"> <li>• <b>ADISK</b> - Local disk currently available to batch jobs in MB.</li> <li>• <b>AMEM</b> - Real memory currently available to batch jobs in MB.</li> <li>• <b>APROCS</b> - Processors currently available to batch jobs on node (configured procs - dedicated procs).</li> <li>• <b>ARCH[&lt;ARCH&gt;]</b> - Processor architecture.</li> <li>• <b>ASWAP</b> - Virtual memory currently available to batch jobs in MB.</li> <li>• <b>CDISK</b> - Total local disk allocated for use by batch jobs in MB.</li> <li>• <b>CMEM</b> - Total real memory on node in MB.</li> <li>• <b>COST</b> - Based on node <b>CHARGERATE</b>.</li> <li>• <b>CPROCS</b> - Total processors on node.</li> <li>• <b>CSWAP</b> - Total virtual memory configured on node in MB.</li> <li>• <b>FEATURE[&lt;FNAME&gt;]</b> - Boolean; specified feature is present on node.</li> <li>• <b>FREETIME</b> - FREETIME is calculated as the time during which there is no reservation on the machine. It uses either the job wallclock limit (if there is a job), or 2 months. The more free time a node has within either the job wallclock limit or 2 months, the higher this value will be.</li> <li>• <b>GMETRIC[&lt;GMNAME&gt;]</b> - Current value of specified generic metric on node.</li> <li>• <b>JOB COUNT</b> - Number of jobs currently running on node.</li> <li>• <b>JOB FREETIME</b> - The number of seconds that the node is idle between now and when the job is scheduled to start.</li> <li>• <b>LOAD</b> - Current 1 minute load average.</li> <li>• <b>MTBF</b> - Mean time between failures (in seconds).</li> <li>• <b>NODEINDEX</b> - Node's nodeindex as specified by the resource manager.</li> <li>• <b>OS</b> - True if job compute requirements match node operating system.</li> <li>• <b>PARAPROCS</b> - Processors currently available to batch jobs within partition (configured procs - dedicated procs).</li> <li>• <b>POWER</b> - TRUE if node is ON.</li> <li>• <b>PREF</b> - Boolean; node meets job specific resource preferences.</li> <li>• <b>PRIORITY</b> - Administrator specified node priority.</li> <li>• <b>RANDOM</b> - Per iteration random value between 0 and 1. (Allows introduction of random allocation factor.)</li> </ul>

Allocation algorithm name	Alias	Description
		<p><i>Example 5:</i> Pack tasks onto nodes with the most processors available and the lowest CPU temperature.</p> <pre> RMCFG[torque] TYPE=pbs RMCFG[temp] TYPE=NATIVE CLUSTERQUERYURL=exec://\$TOOLS DIR/hwmon.pl NODEALLOCATIONPOLICY PRIORITY NODECFG[DEFAULT] PRIORITYF='100*APROCS - GMETRIC[temp] ... </pre>
<b>MINRESOURCE</b>	Min-imumCon-figuredResources	Prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources that still meet the job's resource constraints are selected.
<b>CONTIGUOUS</b>	Contiguous	Allocates nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	ProcessorSpeedBalance	Attempts to allocate the most balanced set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node procspeed. Thus, if possible, nodes with identical procspeeds are allocated to the job. If identical procspeed nodes cannot be found, the algorithm allocates the set of nodes with the minimum node procspeed span or range.

## User-Defined Algorithms

User-defined algorithms allow administrators to define their own algorithms based on factors such as their system's network topology. When node allocation is based on topology, jobs finish faster, administrators see better cluster productivity and users pay less for resources.

### PLUGIN

This algorithm allows administrators to define their own node allocation policy and create a plug-in that allocates nodes based on factors such as a cluster's network topology. This has the following advantages:

- plug-ins keep the source code of the cluster's interconnect network for node allocation separate from Moab's source code (customers can implement plug-ins independent of Moab's release schedule)
- plug-ins can be independently created and tailored to specific hardware and network topology
- plug-ins can be modified without assistance from Adaptive Computing, Inc.

## Specifying *Per Job* Resource Preferences

While the resource based node allocation algorithms can make a good guess at what compute resources would best satisfy a job, sites often possess a subset of jobs that benefit from more explicit resource allocation specification. For example one job may perform best on a particular subset of nodes due to direct access to a tape drive, another may be very memory intensive. Resource preferences are distinct from node requirements. While the former describes what a job needs to run at all, the latter describes what the job needs to run well. In general, a scheduler must satisfy a job's node requirement specification and then satisfy the job's resource preferences as well as possible.

### Specifying Resource Preferences

A number of resource managers natively support the concept of resource preferences (such as Loadleveler). When using these systems, the language specific preferences keywords may be used. For systems that do not support resource preferences natively, Moab provides a [resource manager extension](#) keyword, "[PREF](#)," which you can use to specify desired resources. This extension allows specification of node features, memory, swap, and disk space conditions that define whether the node is considered preferred.

 Moab 5.2 (and earlier) only supports feature-based preferences.

### Selecting Preferred Resources

Enforcing resource preferences is not completely straightforward. A site may have a number of potentially conflicting requirements that the scheduler is asked to simultaneously satisfy. For example, a scheduler may be asked to maximize the proximity of the allocated nodes at the same time it is supposed to satisfy resource preferences and minimize node overcommitment. To allow site specific weighting of these varying requirements, Moab allows resource preferences to be enabled through the [PRIORITY](#) node allocation algorithm. For example, to use resource preferences together with node load, the following configuration might be used:

```
NODEALLOCATIONPOLICY PRIORITY
NODECFG[DEFAULT]    PRIORITYF='5 * PREF - LOAD'
...
```

To request specific resource preferences, a user could then submit a job indicating those preferences. In the case of a PBS job, the following can be used:

```
> qsub -l nodes=4,walltime=1:00:00,pref=feature:fast
```

#### Related topics

- [Generic Metrics](#)
- Per Job Node Allocation Policy Specification via [Resource Manager Extensions](#)

## Node Access Policies

Moab allocates resources to jobs on the basis of a job task—an atomic collection of resources that must be co-located on a single compute node. A given job may request 20 tasks where each task is defined as one processor and 128 MB of RAM. Compute nodes with multiple processors often possess enough resources to support more than one task simultaneously. When it is possible for more than one task to run on a node, node access policies determine which tasks may share the compute node's resources.

Moab supports a distinct number of node access policies that are listed in the following table:

Policy	Description
<b>SHARED</b>	Tasks from any combination of jobs may use available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may use available resources.
<b>SINGLEACCOUNT</b>	Tasks from any jobs owned by the same account may use available resources.
<b>SINGLEGROUP</b>	Tasks from any jobs owned by the same group may use available resources.
<b>SINGLEJOB</b>	Only tasks from a single job may use the node's resources.
<b>SINGLETASK</b>	Only a single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may use available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node. <b>UNIQUEUSER</b> limits the number of jobs a single user can run on a node, allowing other users to run jobs with the remaining resources.

 This policy is useful in environments where job epilog/prologs scripts are used to clean up processes based on userid.

## Configuring Node Access Policies

The global node access policies may be specified via the parameter [NODEACCESSPOLICY](#). This global default may be overridden on a per node basis with the [ACCESS](#) attribute of the [NODECFG](#) parameter or on a per job basis using the resource manager extension [NACCESSPOLICY](#). Finally, a per queue node access policy may also be specified by setting either the [NODEACCESSPOLICY](#) or [FORCENODEACCESSPOLICY](#) attributes of the [CLASSCFG](#) parameter. **FORCENODEACCESSPOLICY** overrides any per job specification in all cases, whereas **NODEACCESSPOLICY** is overridden by per job specification.

By default, nodes are accessible using the setting of the system wide **NODEACCESSPOLICY** parameter unless a specific **ACCESS** policy is specified on a per node basis using the **NODECFG** parameter. Jobs may override this policy and subsequent jobs are bound to conform to the access policies of all jobs currently

running on a given node. For example, if the **NODEACCESSPOLICY** parameter is set to **SHARED**, a new job may be launched on an idle node with a job specific access policy of **SINGLEUSER**. While this job runs, the effective node access policy changes to **SINGLEUSER** and subsequent job tasks may only be launched on this node provided they are submitted by the same user. When all single user jobs have completed on that node, the effective node access policy reverts back to **SHARED** and the node can again be used in **SHARED** mode.

For example, to set a global policy of **SINGLETASK** on all nodes except nodes 13 and 14, use the following:

```
# by default, enforce dedicated node access on all nodes
NODEACCESSPOLICY SINGLETASK
# allow nodes 13 and 14 to be shared
NODECFG[node13] ACCESS=SHARED
NODECFG[node14] ACCESS=SHARED
```

Related topics

- Per job [accesspolicy](#) specification via [Resource Manager Extensions](#)
- [JOBNODEMATCHPOLICY](#) parameter
- [NODEAVAILABILITY](#) parameter

## Node Availability Policies

- [Node Resource Availability Policies](#)
- [Node Categorization](#)
- [Node Failure/Performance Based Notification](#)
- [Node Failure/Performance Based Triggers](#)
- [Handling Transient Node Failures](#)
- [Allocated Resource Failure Policy for Jobs on page 409](#)

Moab enables several features relating to node availability. These include policies that determine how per node resource availability should be reported, how node failures are detected, and what should be done in the event of a node failure.

### Node Resource Availability Policies

Moab allows a job to be launched on a given compute node as long as the node is not full or busy. The [NODEAVAILABILITYPOLICY](#) parameter allows a site to determine what criteria constitute a node being busy. The legal settings are listed in the following table:

Availability Policy	Description
<b>DEDICATED</b>	The node is considered busy if dedicated resources equal or exceed configured resources.

Availability Policy	Description
<b>UTILIZED</b>	The node is considered busy if utilized resources equal or exceed configured resources.
<b>COMBINED</b>	The node is considered busy if either dedicated or utilized resources equal or exceed configured resources.

The default setting for all nodes is **COMBINED**, indicating that a node can accept workload so long as the jobs that the node was allocated to do not request or use more resources than the node has available. In a load balancing environment, this may not be the desired behavior. Setting the **NODEAVAILABILITYPOLICY** parameter to **UTILIZED** allows jobs to be packed onto a node even if the aggregate resources requested exceed the resources configured. For example, assume a scenario with a 4-processor compute node and 8 jobs requesting 1 processor each. If the resource availability policy was set to **COMBINED**, this node would only allow 4 jobs to start on this node even if the jobs induced a load of less than 1.0 each. With the resource availability policy set to **UTILIZED**, the scheduler continues allowing jobs to start on the node until the node's load average exceeds a per processor load value of 1.0 (in this case, a total load of 4.0). To prevent a node from being over populated within a single scheduling iteration, Moab artificially raises the node's load for one scheduling iteration when starting a new job. On subsequent iterations, the actual measured node load information is used.

### Per Resource Availability Policies

By default, the **NODEAVAILABILITYPOLICY** sets a global per node resource availability policy. This policy applies to all resource types on each node such as processors, memory, swap, and local disk. However, the syntax of this parameter is as follows:

```
<POLICY> [ :<RESOURCETYPE> ] . . .
```

This syntax allows per resource availability specification. For example, consider the following:

```
NODEAVAILABILITYPOLICY DEDICATED:PROC COMBINED:MEM COMBINED:DISK
. . .
```

This configuration causes Moab to only consider the quantity of processing resources actually dedicated to active jobs running on each node and ignore utilized processor information (such as CPU load). For memory and disk, both utilized resource information and dedicated resource information should be combined to determine what resources are actually available for new jobs.

### Node Categorization

Moab allows organizations to detect and use far richer information regarding node status than the standard batch "idle," "busy," "down states" commonly found. Using node categorization, organizations can record, track, and report on per node and cluster level status including the following categories:

Category	Description
<b>Active</b>	Node is healthy and currently executing batch workload.
<b>BatchFailure</b>	Node is unavailable due to a failure in the underlying batch system (such as a resource manager server or resource manager node daemon).
<b>Benchmark</b>	Node is reserved for benchmarking.
<b>EmergencyMaintenance</b>	Node is reserved for unscheduled system maintenance.
<b>GridReservation</b>	Node is reserved for grid use.
<b>HardwareFailure</b>	Node is unavailable due to a failure in one or more aspects of its hardware configuration (such as a power failure, excessive temperature, memory, processor, or swap failure).
<b>HardwareMaintenance</b>	Node is reserved for scheduled system maintenance.
<b>Idle</b>	Node is healthy and is currently not executing batch workload.
<b>JobReservation</b>	Node is reserved for job use.
<b>NetworkFailure</b>	Node is unavailable due to a failure in its network adapter or in the switch.
<b>Other</b>	Node is in an uncategorized state.
<b>OtherFailure</b>	Node is unavailable due to a general failure.
<b>PersonalReservation</b>	Node is reserved for dedicated use by a personal reservation.
<b>Site[1-8]</b>	Site specified usage categorization.
<b>SoftwareFailure</b>	Node is unavailable due to a failure in a local software service (such as automounter, security or information service such as NIS, local databases, or other required software services).
<b>SoftwareMaintenance</b>	Node is reserved for software maintenance.
<b>StandingReservation</b>	Node is reserved by a standing reservation.

Category	Description
<b>StorageFailure</b>	Node is unavailable due to a failure in the cluster storage system or local storage infrastructure (such as failures in Lustre, GPFS, PVFS, or SAN).
<b>UserReservation</b>	Node is reserved for dedicated use by a particular user or group and may or may not be actively executing jobs.

Node categories can be explicitly assigned by cluster administrators using the [mrsvctl -c](#) command to create a reservation and associate a category with that node for a specified timeframe. Further, outside of this explicit specification, Moab automatically mines all configured interfaces to learn about its environment and the health of the resources it is managing. Consequently, Moab can identify many hardware failures, software failures, and batch failures without any additional configuration. However, it is often desirable to make additional information available to Moab to allow it to integrate this information into reports; automatically notify managers, users, and administrators; adjust internal policies to steer workload around failures; and launch various custom [triggers](#) to rectify or mitigate the problem.

**i** You can specify the [FORCERSVSUBTYPE](#) parameter to require all administrative reservations be associated with a node category at reservation creation time. For example:

```
NODECFG [DEFAULT] ENABLEPROFILING=TRUE
FORCERSVSUBTYPE TRUE
```

Node health and performance information from external systems can be imported into Moab using the [native resource manager interface](#). This is commonly done using [generic metrics](#) or [consumable generic resources](#) for performance and node categories or node variables for status information. Combined with arbitrary node messaging information, Moab can combine detailed information from remote services and report this to other external services.

**i** Use the [NODECATCREDLIST](#) parameter to generate extended node category based statistics.

## Node Failure/Performance Based Notification

Moab can be configured to cause node failures and node performance levels that cross specified thresholds to trigger notification events. This is accomplished using the [GEVENTCFG](#) parameter as described in the [Generic Event Overview](#) section. For example, the following configuration can be used to trigger an email to administrators each time a node is marked down.

```
GEVENTCFG [nodedown] ACTION=notify REARM=00:20:00
...
```

## Node Failure/Performance Based Triggers

Moab supports per node triggers that can be configured to fire when specific events are fired or specific thresholds are met. These triggers can be used to modify internal policies or take external actions. A few

examples follow:

- decrease node allocation priority if node throughput drops below threshold X
- launch local diagnostic/recovery script if parallel file system mounts become stale
- reset high performance network adapters if high speed network connectivity fails
- create general system reservation on node if processor or memory failure occurs

As mentioned, Moab triggers can be used to initiate almost any action, from sending mail to updating a database, to publishing data for an SNMP trap, to driving a web service.

## Handling Transient Node Failures

Since Moab actively schedules both current and future actions of the cluster, it is often important for it to have a reasonable estimate of when failed nodes will be again available for use. This knowledge is particularly useful for proper scheduling of new jobs and management of resources in regard to [backfill](#). With backfill, Moab determines which resources are available for priority jobs and when the highest priority idle jobs can run. If a node experiences a failure, Moab should have a concept of when this node will be restored.

When Moab analyzes [down](#) nodes for allocation, one of two issues may occur with the highest priority jobs. If Moab believes that down nodes will not be recovered for an extended period of time, a transient node failure within a reservation for a priority job may cause the reservation to slide far into the future allowing other lower priority jobs to allocate and launch on nodes previously reserved for it. Moments later, when the transient node failures are resolved, Moab may be unable to restore the early reservation start time as other jobs may already have been launched on previously available nodes.

In the reverse scenario, if Moab recognizes a likelihood that down nodes will be restored too quickly, it may make reservations for top priority jobs that allocate those nodes. Over time, Moab slides those reservations further into the future as it determines that the reserved nodes are not being recovered. While this does not delay the start of the top priority jobs, these unfulfilled reservations can end up blocking other jobs that should have properly been backfilled and executed.

## Creating Automatic Reservations

If a node experiences occasional transient failures (often not associated with a node state of down), Moab can automatically create a temporary reservation over the node to allow the transient failure time to clear and prevent Moab from attempting to re-use the node while the failure is active. This reservation behavior is controlled using the [NODEFAILURERESERVETIME](#) parameter as in the following example:

```
# reserve nodes for 1 minute if transient failures are detected
NODEFAILURERESERVETIME 00:01:00
```

## Blocking Out Down Nodes

If one or more resource managers identify failures and mark nodes as down, Moab can be configured to associate a default *unavailability* time with this failure and the node state *down*. This is accomplished using the [NODEDOWNSTATEDELAYTIME](#) parameter. This delay time floats and is measured as a fixed time into the future from the time "NOW"; it is not associated with the time the node was originally

marked down. For example, if the delay time was set to 10 minutes, and a node was marked down 20 minutes ago, Moab would still consider the node unavailable until 10 minutes into the future.

While it is difficult to select a good default value that works for all clusters, the following is a general rule of thumb:

- Increase **NODEDOWNSTATEDELAYTIME** if jobs are getting blocked due to priority reservations sliding as down nodes are not recovered.
- Decrease **NODEDOWNSTATEDELAYTIME** if high priority job reservations are getting regularly delayed due to transient node failures.

```
# assume down nodes will not be recovered for one hour
NODEDOWNSTATEDELAYTIME 01:00:00
```

## Allocated Resource Failure Policy for Jobs

If a failure occurs within a collection of nodes allocated to a job, Moab can automatically re-allocate replacement resources. This can be configured with [JOBACTIONONNODEFAILURE](#).

How an active job behaves when one or more of its allocated resources fail depends on the allocated resource failure policy. Depending on the type of job, type of resources, and type of middleware infrastructure, a site may choose to have different responses based on the job, the resource, and the type of failure.

### Failure Responses

By default, Moab cancels a job when an allocated resource failure is detected. However, you can specify the following actions:

Option	Policy action
<b>CANCEL</b>	Cancels the job
<b>FAIL</b>	Terminates the job as a failed job
<b>HOLD</b>	Places a hold on the job. This option is only applicable if you are using checkpointing
<b>IGNORE</b>	Ignores the failed node, allowing the job to proceed
<b>NOTIFY</b>	Notifies the administrator and user of failure but takes no further action
<b>REQUEUE</b>	Requeues job and allows it to run when alternate resources become available

### Policy Precedence

For a given job, the applied policy can be set at various levels with policy precedence applied in the job, class/queue, partition, and then system level. The following table indicates the available methods for

setting this policy:

Object	Parameter	Example
Job	<a href="#">RESFAILPOLICY</a> resource manager extension	<pre>&gt; qsub -l resfailpolicy=requeue</pre>
Class/Queue	<a href="#">RESFAILPOLICY</a> attribute of <a href="#">CLASSCFG</a> parameter	<pre>CLASSCFG[batch] RESFAILPOLICY=CANCEL</pre>
Partition	<a href="#">JOBACTIONNONNODEFAILURE</a> attribute of <a href="#">PARCFG</a> parameter	<pre>PARCFG[web3] JOBACTIONNONNODEFAILURE=NOTIFY</pre>
System	<a href="#">NODEALLOCRESFAILUREPOLICY</a> parameter	<pre>NODEALLOCRESFAILUREPOLICY=MIGRATE</pre>

## Failure Definition

Any allocated node going down constitutes a failure. However, for certain types of workload, responses to failures may be different depending on whether it is the master task (task 0) or a slave task that fails. To indicate that the associated policy should only take effect if the master task fails, the allocated resource failure policy should be specified with a trailing asterisk (\*), as in the following example:

```
CLASSCFG[virtual_services] RESFAILPOLICY=requeue*
```

## TORQUE Failure Details

When a node fails becoming unresponsive, the resource manager central daemon identifies this failure within a configurable time frame (default: 60 seconds). Detection of this failure triggers an event that causes Moab to immediately respond. Based on the specified policy, Moab notifies administrators, holds the job, requeues the job, allocates replacement resources to the job, or cancels the job. If the job is canceled or requeued, Moab sends the request to TORQUE, which immediately frees all non-failed resources making them available for use by other jobs. Once the failed node is recovered, it contacts the resource manager central daemon, determines that the associated job has been canceled/requeued, cleans up, and makes itself available for new workload.

### Related topics

- [Node State Overview](#)
- [JOBACTIONNONNODEFAILURE](#) parameter
- [NODEFAILURERESERVETIME](#) parameter
- [NODEDOWNSTATEDELAYTIME](#) parameter (down nodes will be marked unavailable for the specified duration)
- [NODEDRAINSTATEDELAYTIME](#) parameter (offline nodes will be marked unavailable for the specified duration)
- [NODEBUSYSTATEDELAYTIME](#) parameter (nodes with unexpected background load will be marked unavailable for the specified duration)

- [NODEALLOCRESFAILUREPOLICY](#) parameter (action to take if executing jobs have one or more allocated nodes fail)

## Task Distribution Policies

Under Moab, task distribution policies are specified at a global scheduler level, a global resource manager level, or at a per job level. In addition, you can set up some aspects of task distribution as defaults on a per class basis.

Related topics

- [Node Set Overview](#)
- [Node Allocation Overview](#)

## Managing Fairness - Throttling Policies, Fairshare, and Allocation Management

- [Fairness Overview on page 411](#)
- [Usage Limits/Throttling Policies on page 414](#)
- [Fairshare on page 432](#)
- [Charging a Workflow on page 446](#)

## Fairness Overview

The concept of cluster fairness varies widely from person to person and site to site. While some interpret it as giving all users equal access to compute resources, more complicated concepts incorporating historical resource usage, political issues, and job value are equally valid. While no scheduler can address all possible definitions of fair, Moab provides one of the industry's most comprehensive and flexible set of tools allowing most sites the ability to address their many and varied fairness management needs.

Under Moab, most fairness policies are addressed by a combination of the facilities described in the following table:

Job Prioritization	
Description:	Specifies what is most important to the scheduler. Using service based priority factors allows a site to balance job turnaround time, expansion factor, or other scheduling performance metrics.

### Job Prioritization

Example:

```
SERVICEWEIGHT 1
QUEUETIMEWEIGHT 10
```

*Causes jobs to increase in priority by 10 points for every minute they remain in the queue.*

### Usage Limits (Throttling Policies)

Description:

Specifies limits on exactly what resources can be used at any given instant.

Example:

```
USERCFG[john] MAXJOB=3
GROUPCFG[DEFAULT] MAXPROC=64
GROUPCFG[staff] MAXPROC=128
```

*Allows john to only run 3 jobs at a time. Allows the group staff to use up to 128 total processors and all other groups to use up to 64 processors.*

### Fairshare

Description:

Specifies usage targets to limit resource access or adjust priority based on historical cluster and grid level resource usage.

Example:

```
USERCFG[steve] FSTARGET=25.0+
FSWEIGHT 1
FSUSERWEIGHT 10
```

*Enables priority based fairshare and specifies a fairshare target for user steve such that his jobs are favored in an attempt to keep his jobs using at least 25.0% of delivered compute cycles.*

### Allocation Management

Description:

Specifies long term, credential-based resource usage limits.

Example:

```
AMCFG[mam] TYPE=MAM HOST=server.sys.net
```

*Enables the Moab Accounting Manager allocation management interface. Within the allocation manager, project or account based allocations may be configured. These allocations may, for example, do such things as allow project X to use up to 100,000 processor-hours per quarter, provide various QoS sensitive charge rates, and share allocation access.*

Quality of Service	
Description:	Specifies additional resource and service access for particular users, groups, and accounts. QoS facilities can provide special priorities, policy exemptions, reservation access, and other benefits (as well as special charge rates).
Example:	<pre>QOSCFG[orion] PRIORITY=1000 XFTARGET=1.2 QOSCFG[orion] QFLAGS=PREEMPTOR,IGNSYSTEM,RESERVEALWAYS</pre> <p><i>Enables jobs requesting the orion QoS a priority increase, an expansion factor target to improve response time, the ability to preempt other jobs, an exemption from system level job size policies, and the ability to always reserve needed resources if it cannot start immediately.</i></p>

Standing Reservations	
Description:	Reserves blocks of resources within the cluster for specific, periodic time frames under the constraints of a flexible access control list.
Example:	<pre>SRCFG[jupiter] HOSTLIST=node01[1-4] SRCFG[jupiter] STARTTIME=9:00:00 ENDTIME=17:00:00 SRCFG[jupiter] USERLIST=john,steve ACCOUNTLIST=jupiter</pre> <p><i>Reserve nodes <b>node011</b> through <b>node014</b> from 9:00 AM until 5:00 PM for use by jobs from user john or steve or from the project jupiter.</i></p>

Class/Queue Constraints	
Description:	Associates users, resources, priorities, and limits with cluster classes or cluster queues that can be assigned to or selected by end-users.
Example:	<pre>CLASSCFG[long] MIN.WCLIMIT=24:00:00 SRCFG[jupiter] PRIORITY=10000 SRCFG[jupiter] HOSTLIST=acn[1-4][0-9]</pre> <p><i>Assigns long jobs a high priority but only allow them to run on certain nodes.</i></p>

## Selecting the Correct Policy Approach

Moab supports a rich set of policy controls in some cases allowing a particular policy to be enforced in more than one way. For example, cycle distribution can be controlled using usage limits, fairshare, or even queue definitions. Selecting the most correct policy depends on site objectives and needs; consider the following when making such a decision:

- Minimal end-user training
  - Does the solution use an approach familiar to or easily learned by existing users?
- End-user transparency
  - Can the configuration be enabled or disabled without impacting user behavior or job submission?
- Impact on system utilization and system responsiveness
- Solution complexity
  - Is the impact of the configuration readily intuitive, and is it easy to identify possible side effects?
- Solution extensibility and flexibility
  - Will the proposed approach allow the solution to be easily tuned and extended as cluster needs evolve?

#### Related topics

- [Job Prioritization](#)
- [Usage Limits \(Throttling Policies\)](#)
- [Fairshare](#)
- Allocation Management
- [Quality of Service](#)
- [Standing Reservations](#)
- [Class/Queue Constraints](#)

## Usage Limits/Throttling Policies

A number of Moab policies allow an administrator to control job flow through the system. These throttling policies work as filters allowing or disallowing a job to be considered for scheduling by specifying limits regarding system usage for any given moment. These policies may be specified as global or specific constraints specified on a per user, group, account, QoS, or class basis.

- [Fairness via Throttling Policies](#)
  - [Basic Fairness Policies](#)
  - [Multi-Dimension Fairness Policies](#)
- [Override Limits](#)
- [Idle Job Limits](#)
- [Hard and Soft Limits](#)
- [Per-partition Limits](#)

- [Usage-based limits on page 429](#)
  - [Configuring Actions on page 429](#)
  - [Specifying Hard and Soft Policy Violations on page 431](#)
  - [Constraining Walltime Usage on page 432](#)

## Fairness via Throttling Policies

Moab allows significant flexibility with usage limits, or throttling policies. At a high level, Moab allows resource usage limits to be specified in three primary workload categories: (1) active, (2) idle, and (3) system job limits.

### Basic Fairness Policies

Workload category	Description
<b>Active job limits</b>	Constrain the total cumulative resources available to active jobs at a given time.
<a href="#">Idle job limits</a>	Constrain the total cumulative resources available to idle jobs at a given time.
<b>System job limits</b>	Constrain the maximum resource requirements of any single job.

These limits can be applied to any job credential (user, group, account, QoS, and class), or on a system-wide basis. Using the keyword *DEFAULT*, a site may also specify the default setting for the desired user, group, account, QoS, and class. Additionally, you may configure QoS to allow limit overrides to any particular policy.

To run, a job must meet all policy limits. Limits are applied using the **\*CFG** set of parameters, particularly [USERCFG](#), [GROUPCFG](#), [ACCOUNTCFG](#), [QOSCFG](#), [CLASSCFG](#), and [SYSCFG](#). Limits are specified by associating the desired limit to the individual or default object. The usage limits currently supported are listed in the following table.

MAXARRAYJOB	
<b>Units</b>	Number of simultaneous active array job sub-jobs.
<b>Description</b>	Limits the number of simultaneously active (starting or running) array sub-jobs a credential can have.
<b>Example</b>	<pre>USERCFG[gertrude] MAXARRAYJOB=10</pre> <p><i>Gertrude can have a maximum of 10 active job array sub-jobs.</i></p>

MAXGRES	
<b>Units</b>	# of concurrent uses of a generic resource
<b>Description</b>	Limits the concurrent usage of a generic resource to a specific quantity or quantity range.
<b>Example</b>	<pre>USERCFG[joe] MAXGRES[matlab]=2 USERCFG[jim] MAXGRES[matlab]=2,4</pre>

MAXJOB	
<b>Units</b>	# of jobs
<b>Description</b>	<p>Limits the number of jobs a credential may have active (starting or running) at any given time. Moab places a hold on all new jobs submitted by that credential once it has reached its maximum number of allowable jobs.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin: 5px 0;"> <p><b>i</b> MAXJOB=0 is not supported. You can, however, achieve similar results by using the <a href="#">HOLD</a> attribute of the <a href="#">USERCFG</a> parameter:</p> <pre>USERCFG[john] HOLD=yes</pre> </div>
<b>Example</b>	<pre>USERCFG[DEFAULT] MAXJOB=8 GROUPCFG[staff] MAXJOB=2,4</pre>

MAXMEM	
<b>Units</b>	total memory in MB
<b>Description</b>	Limits the total amount of dedicated memory (in MB) that can be allocated by a credential's active jobs at any given time.
<b>Example</b>	<pre>ACCOUNTCFG[jasper] MAXMEM=2048</pre>

MAXNODE	
<b>Units</b>	# of nodes

MAXNODE	
<b>Description</b>	Limits the total number of compute nodes that can be in use by active jobs at any given time. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> On some systems (including TORQUE/PBS), nodes have been softly defined rather than strictly defined; that is, a job may request 2 nodes but TORQUE will translate this request into 1 node with 2 processors. This can prevent Moab from enforcing a <b>MAXNODE</b> policy correctly for a single job. Correct behavior can be achieved using <b>MAXPROC</b>.</p> </div>
<b>Example</b>	<code>CLASSCFG[batch] MAXNODE=64</code>

MAXPE	
<b>Units</b>	# of <a href="#">processor equivalents</a>
<b>Description</b>	Limits the total number of dedicated processor-equivalents that can be allocated by active jobs at any given time.
<b>Example</b>	<code>QOSCFG[base] MAXPE=128</code>

MAXPROC	
<b>Units</b>	# of processors
<b>Description</b>	Limits the total number of dedicated processors that can be allocated by active jobs at any given time per credential. To set <b>MAXPROC</b> per job, use <a href="#">msub -W</a> .
<b>Example</b>	<code>CLASSCFG[debug] MAXPROC=32</code>

MAXPS	
<b>Units</b>	<# of processors> * <walltime>
<b>Description</b>	Limits the number of outstanding processor-seconds a credential may have allocated at any given time. For example, if a user has a 4-processor job that will complete in 1 hour and a 2-processor job that will complete in 6 hours, they have $4 * 1 * 3600 + 2 * 6 * 3600 = 16 * 3600$ outstanding processor-seconds. The outstanding processor-second usage of each credential is updated each scheduling iteration, decreasing as jobs approach their completion time.

MAXPS	
<b>Example</b>	<code>USERCFG[DEFAULT] MAXPS=720000</code>
MAXSUBMITJOBS	
<b>Units</b>	# of jobs
<b>Description</b>	Limits the number of jobs a credential may submit and have in the system at once. Moab will reject any job submitted beyond this limit. If you use a TORQUE resource manager, you should also set <code>max_user_queueable</code> in case the user submits jobs via <code>qsub</code> instead of <code>msub</code> . See " <a href="#">Queue attributes on page 2626</a> " in the <i>TORQUE Administrator Guide</i> for more information.
<b>Example</b>	<code>USERCFG[DEFAULT] MAXSUBMITJOBS=5</code>
MAXWC	
<b>Units</b>	job duration [[[DD:]HH:]MM:]SS
<b>Description</b>	Limits the cumulative remaining walltime a credential may have associated with active jobs. It behaves identically to the <a href="#">MAXPS on page 417</a> limit (listed earlier) only lacking the processor weighting. Like MAXPS, the cumulative remaining walltime of each credential is also updated each scheduling iteration.  <div style="border: 1px solid #0056b3; padding: 5px; background-color: #e6f2ff;"><b>i</b> MAXWC does not limit the maximum wallclock limit per job. For this capability, use <a href="#">MAX.WCLIMIT on page 137</a>.</div>
<b>Example</b>	<code>USERCFG[ops] MAXWC=72:00:00</code>

The following example demonstrates a simple limit specification:

<code>USERCFG[DEFAULT] MAXJOB=4</code>
<code>USERCFG[john] MAXJOB=8</code>
<i>This example allows user john to run up to 8 jobs while all other users may only run up to 4.</i>

Simultaneous limits of different types may be applied per credential and multiple types of credentials may have limits specified. The next example demonstrates this mixing of limits and is a bit more complicated.

```

USERCFG[steve]      MAXJOB=2  MAXNODE=30
GROUPCFG[staff]    MAXJOB=5
CLASSCFG[DEFAULT]  MAXNODE=16
CLASSCFG[batch]    MAXNODE=32

```

This configuration may potentially apply multiple limits to a single job. As discussed previously, a job may only run if it satisfies all applicable limits. Thus, in this example, the scheduler will be constrained to allow at most **2** simultaneous user *steve* jobs with an aggregate node consumption of no more than **30** nodes. However, if the job is submitted to a class other than *batch*, it may be limited further. Here, only **16** total nodes may be used simultaneously by jobs running in any given class with the exception of the class *batch*. If *steve* submitted a job to run in the class *interactive*, for example, and there were jobs already running in this class using a total of 14 nodes, his job would be blocked unless it requested 2 or fewer nodes by the default limit of **16** nodes per class.

## Multi-Dimension Fairness Policies and Per Credential Overrides

Multi-dimensional fairness policies allow a site to specify policies based on combinations of job credentials. A common example might be setting a maximum number of jobs allowed per queue per user or a total number of processors per group per QoS. As with basic fairness policies, multi-dimension policies are specified using the **\*CFG** parameters or through the [identity manager interface](#). Moab supports the most commonly used multi-dimensional fairness policies (listed in the table below) using the following format:

```
*CFG [X] <LIMITTYPE>[<CRED>]=<LIMITVALUE>
```

**\*CFG** is one of **USERCFG**, **GROUPCFG**, **ACCOUNTCFG**, **QOSCFG**, or **CLASSCFG**, the **<LIMITTYPE>** policy is one of the policies listed in the table in section [6.2.1.1](#), and **<CRED>** is of the format **<CREDTYPE>[:<VALUE>]** with **CREDTYPE** being one of **USER**, **GROUP**, **ACCT**, **QoS**, or **CLASS**. The optional **<VALUE>** setting can be used to specify that the policy only applies to a specific credential value. For example, the following configuration sets limits on the class *fast*, controlling the maximum number of jobs any group can have active at any given time and the number of processors in use at any given time for user *steve*.

```

CLASSCFG[fast] MAXJOB[GROUP]=12
CLASSCFG[fast] MAXPROC[USER:steve]=50
CLASSCFG[fast] MAXIJOB[USER]=10

```

The following example configuration may clarify further:

```

# allow class batch to run up the 3 simultaneous jobs
# allow any user to use up to 8 total nodes within class
CLASSCFG[batch] MAXJOB=3 MAXNODE[USER]=8
# allow users steve and bob to use up to 3 and 4 total processors respectively within
class
CLASSCFG[fast] MAXPROC[USER:steve]=3 MAXPROC[USER:bob]=4

```



Multi-dimensional policies cannot be applied on **DEFAULT** credentials.

The table below lists the currently implemented, multi-dimensional usage limit permutations. The "slmt" stands for "Soft Limit" and "hlmt" stands for "Hard Limit."

Multi-dimension usage limit permutations	
ACCOUNTCFG[name]	MAXIJOB[QOS]=hlmt MAXIJOB[QOS:qosname]=hlmt
	MAXIPROC[QOS]=hlmt MAXIPROC[QOS:qosname]=hlmt
	MAXJOB[QOS]=slmt,hlmt MAXJOB[QOS:qosname]=slmt,hlmt
	MAXJOB[USER]=slmt,hlmt MAXJOB[USER:username]=slmt,hlmt
	MAXMEM[USER]=slmt,hlmt MAXMEM[USER:username]=slmt,hlmt
	MAXNODE[USER]=slmt,hlmt MAXNODE[USER:username]=slmt,hlmt
	MAXPE[QOS]=slmt,hlmt MAXPE[QOS:qosname]=slmt,hlmt
	MAXPROC[USER]=slmt,hlmt MAXPROC[USER:username]=slmt,hlmt
	MAXPROC[QOS]=slmt,hlmt MAXPROC[QOS:qosname]=slmt,hlmt
	MAXPROC[USER]=slmt,hlmt MAXPROC[USER:username]=slmt,hlmt
	MAXPS[QOS]=slmt,hlmt MAXPS[QOS:qosname]=slmt,hlmt
	MAXPS[USER]=slmt,hlmt MAXPS[USER:username]=slmt,hlmt
	MAXWC[USER]=slmt,hlmt MAXWC[USER:username]=slmt,hlmt

Multi-dimension usage limit permutations	
CLASSCFG[name]	MAXJOB[GROUP]=slmt,hlmt MAXJOB[GROUP:groupname]=slmt,hlmt
	MAXJOB[QOS:qosname]=hlmt
	MAXJOB[USER]=slmt,hlmt MAXJOB[USER:username]=slmt,hlmt
	MAXMEM[GROUP]=slmt,hlmt MAXMEM[GROUP]=slmt,hlmt
	MAXMEM[GROUP]=slmt,hlmt MAXMEM[GROUP:groupname]=slmt,hlmt
	MAXMEM[QOS:qosname]=hlmt
	MAXMEM[USER]=slmt,hlmt MAXMEM[USER:username]=slmt,hlmt
	MAXNODE[GROUP]=slmt,hlmt MAXNODE[GROUP:groupname]=slmt,hlmt
	MAXNODE[QOS:qosname]=hlmt
	MAXNODE[USER]=slmt,hlmt MAXNODE[USER:username]=slmt,hlmt
	MAXPE[GROUP]=slmt,hlmt MAXPE[GROUP:groupname]=slmt,hlmt
	MAXPE[QOS:qosname]=hlmt
	MAXPE[USER]=slmt,hlmt MAXPE[USER:username]=slmt,hlmt
	MAXPROC[GROUP]=slmt,hlmt MAXPROC[GROUP:groupname]=slmt,hlmt

### Multi-dimension usage limit permutations

	MAXPROC[QOS:qosname]=hlmt
	MAXPROC[USER]=slmt,hlmt MAXPROC[USER:username]=slmt,hlmt
	MAXPS[GROUP]=slmt,hlmt MAXPS[GROUP:groupname]=slmt,hlmt
	MAXPS[QOS:qosname]=hlmt
	MAXPS[USER]=slmt,hlmt MAXPS[USER:username]=slmt,hlmt
	MAXWC[GROUP]=slmt,hlmt MAXWC[GROUP:groupname]=slmt,hlmt
	MAXWC[QOS:qosname]=hlmt
	MAXWC[USER]=slmt,hlmt MAXWC[USER:username]=slmt,hlmt

Multi-dimension usage limit permutations	
GROUPCFG[name]	MAXJOB[CLASS:classname]=slmt,hlmt
	MAXJOB[USER]=slmt,hlmt MAXJOB[USER:username]=slmt,hlmt
	MAXMEM[CLASS:classname]=slmt,hlmt
	MAXMEM[USER]=slmt,hlmt MAXMEM[USER:username]=slmt,hlmt
	MAXNODE[CLASS:classname]=slmt,hlmt
	MAXNODE[USER]=slmt,hlmt MAXNODE[USER:username]=slmt,hlmt
	MAXPE[CLASS:classname]=slmt,hlmt
	MAXPE[USER]=slmt,hlmt MAXPE[USER:username]=slmt,hlmt
	MAXPROC[CLASS:classname]=slmt,hlmt
	MAXPROC[USER]=slmt,hlmt MAXPROC[USER:username]=slmt,hlmt
	MAXPS[CLASS:classname]=slmt,hlmt
	MAXPS[USER]=slmt,hlmt MAXPS[USER:username]=slmt,hlmt
	MAXWC[CLASS:classname]=slmt,hlmt
	MAXWC[USER]=slmt,hlmt MAXWC[USER:username]=slmt,hlmt

Multi-dimension usage limit permutations	
QOSCFG[name]	MAXIJOB[ACCT]=hlmt MAXIJOB[ACCT:accountname]=hlmt
	MAXIPROC[ACCT]=hlmt MAXIPROC[ACCT:accountname]=hlmt
	MAXJOB[ACCT]=slmt,hlmt MAXJOB[ACCT:accountname]=slmt,hlmt
	MAXJOB[USER]=slmt,hlmt MAXJOB[USER:username]=slmt,hlmt
	MAXMEM[USER]=slmt,hlmt MAXMEM[USER:username]=slmt,hlmt
	MAXNODE[USER]=slmt,hlmt MAXNODE[USER:username]=slmt,hlmt
	MAXPE[ACCT]=slmt,hlmt MAXPE[ACCT:accountname]=slmt,hlmt
	MAXPE[USER]=slmt,hlmt MAXPE[USER:username]=slmt,hlmt
	MAXPROC[ACCT]=slmt,hlmt MAXPROC[ACCT:accountname]=slmt,hlmt
	MAXPROC[USER]=slmt,hlmt MAXPROC[USER:username]=slmt,hlmt
	MAXPS[ACCT]=slmt,hlmt MAXPS[ACCT:accountname]=slmt,hlmt
	MAXPS[USER]=slmt,hlmt MAXPS[USER:username]=slmt,hlmt
	MAXWC[USER]=slmt,hlmt MAXWC[USER:username]=slmt,hlmt

### Multi-dimension usage limit permutations

USERCFG[name]	MAXJOB[GROUP]=slmt,hlmt MAXJOB[GROUP:groupname]=slmt,hlmt
	MAXMEM[GROUP]=slmt,hlmt MAXMEM[GROUP:groupname]=slmt,hlmt
	MAXNODE[GROUP]=slmt,hlmt MAXNODE[GROUP:groupname]=slmt,hlmt
	MAXPE[GROUP]=slmt,hlmt MAXPE[GROUP:groupname]=slmt,hlmt
	MAXPROC[GROUP]=slmt,hlmt MAXPROC[GROUP:groupname]=slmt,hlmt
	MAXPS[GROUP]=slmt,hlmt MAXPS[GROUP:groupname]=slmt,hlmt
	MAXWC[GROUP]=slmt,hlmt MAXWC[GROUP:groupname]=slmt,hlmt

## Override Limits

Like all job credentials, the QoS object may be associated with resource usage limits. However, this credential can also be given special override limits that supersede the limits of other credentials, effectively causing all other limits of the same type to be ignored. See [QoS Usage Limits and Overrides](#) for a complete list of policies that can be overridden. The following configuration provides an example of this in the last line:

```
USERCFG[steve]    MAXJOB=2    MAXNODE=30
GROUPCFG[staff]  MAXJOB=5
CLASSCFG[DEFAULT] MAXNODE=16
CLASSCFG[batch]  MAXNODE=32
QOSCFG[hiprio]   OMAXJOB=3   OMAXNODE=64
```

*Only 3 hiprio QoS jobs may run simultaneously and hiprio QoS jobs may run with up to 64 nodes per credential ignoring other credential **MAXNODE** limits.*

Given the preceding configuration, assume a job is submitted with the credentials, user *steve*, group *staff*, class *batch*, and QoS *hiprio*.

Such a job will start so long as running it does not lead to any of the following conditions:

- Total nodes used by user *steve* does not exceed **64**.
- Total active jobs associated with user *steve* does not exceed **2**.
- Total active jobs associated with group *staff* does not exceed **5**.
- Total nodes dedicated to class *batch* does not exceed **64**.
- Total active jobs associated with QoS *hiprio* does not exceed **3**.

While the preceding example is a bit complicated for most sites, similar combinations may be required to enforce policies found on many systems.

## Idle Job Limits

Idle (or queued) job limits control which jobs are eligible for scheduling. To be eligible for scheduling, a job must meet the following conditions:

- Be idle as far as the resource manager is concerned (no holds).
- Have all job prerequisites satisfied (no outstanding job or data dependencies).
- Meet all idle job throttling policies.

If a job fails to meet any of these conditions, it will not be considered for scheduling and will not accrue service based job prioritization. (See [service component](#) and [JOBPRIOACCRUALPOLICY](#).) The primary purpose of idle job limits is to ensure fairness among competing users by preventing queue stuffing and other similar abuses. Queue stuffing occurs when a single entity submits large numbers of jobs, perhaps thousands, all at once so they begin accruing queue time based priority and remain first to run despite subsequent submissions by other users.

Idle limits are specified in a manner almost identical to active job limits with the insertion of the capital letter *I* into the middle of the limit name. Below are examples of the **MAXIJOB** and **MAXINODE** limits, which are idle limit equivalents to the [MAXJOB on page 416](#) and [MAXNODE on page 416](#) limits:

MAXIJOB	
<b>Units</b>	# of jobs
<b>Description</b>	Limits the number of idle (eligible) jobs a credential may have at any given time.
<b>Example</b>	<pre>USERCFG[DEFAULT]  MAXIJOB=8 GROUPCFG[staff]  MAXIJOB=2,4</pre>

MAXINODE	
<b>Units</b>	# of nodes

MAXINODE	
<b>Description</b>	Limits the total number of compute nodes that can be requested by jobs in the eligible/idle queue at any time. Once the limit is exceeded, the remaining jobs will be placed in the blocked queue. The number of nodes is determined by <code>&lt;tasks&gt; / &lt;maximumProcsOnOneNode&gt;</code> or, if using <a href="#">JOBNODEMATCHPOLICY on page 981</a> <code>EXACTNODE</code> , by the number of nodes requested.
<b>Example</b>	<code>USERCFG[DEFAULT] MAXINODE=2</code>

Idle limits can constrain the total number of jobs considered to be eligible on a per credential basis. Further, like active job limits, idle job limits can also constrain eligible jobs based on aggregate requested resources. This could, for example, allow a site to indicate that for a given user, only jobs requesting up to a total of 64 processors, or 3200 processor-seconds would be considered at any given time. Which jobs to select is accomplished by prioritizing all idle jobs and then adding jobs to the eligible list one at a time in priority order until jobs can no longer be added. This eligible job selection is done only once per scheduling iteration, so, consequently, idle job limits only support a single hard limit specification. Any specified soft limit is ignored.

All single dimensional job limit types supported as active job limits are also supported as idle job limits. In addition, Moab also supports [MAXIJOB\[USER\]](#) and [MAXIPROC\[USER\]](#) policies on a per class basis. (See [Basic Fairness Policies](#).)

**Example:**

<code>USERCFG[steve]</code>	<code>MAXIJOB=2</code>
<code>GROUPCFG[staff]</code>	<code>MAXIJOB=5</code>
<code>CLASSCFG[batch]</code>	<code>MAXIJOB[USER]=2 MAXIJOB[USER:john]=6</code>
<code>QOSCFG[hiprio]</code>	<code>MAXIJOB=3</code>

## Hard and Soft Limits

Hard and soft limit specification allows a site to balance both fairness and utilization on a given system. Typically, throttling limits are used to constrain the quantity of resources a given credential (such as user or group) is allowed to consume. These limits can be very effective in enforcing fair usage among a group of users. However, in a lightly loaded system, or one in which there are significant swings in usage from project to project, these limits can reduce system utilization by blocking jobs even when no competing jobs are queued.

Soft limits help address this problem by providing additional scheduling flexibility. They allow sites to specify two tiers of limits; the more constraining limits soft limits are in effect in heavily loaded situations and reflect tight fairness constraints. The more flexible hard limits specify how flexible the scheduler can be in selecting jobs when there are idle resources available after all jobs meeting the tighter soft limits have started. Soft and hard limits are specified in the format `[<SOFTLIMIT>, ] <HARDLIMIT>`. For example, a given site may want to use the following configuration:

```
USERCFG[DEFAULT] MAXJOB=2, 8
```

*With this configuration, the scheduler would select all jobs that meet the per user **MAXJOB** limit of 2. It would then attempt to start and reserve resources for all of these selected jobs. If after doing so there still remain available resources, the scheduler would then select all jobs that meet the less constraining hard per user **MAXJOB** limit of 8 jobs. These jobs would then be scheduled and reserved as available resources allow.*

*If no soft limit is specified or the soft limit is less constraining than the hard limit, the soft limit is set equal to the hard limit.*

### Example:

```
USERCFG[steve] MAXJOB=2, 4 MAXNODE=15, 30
GROUPCFG[staff] MAXJOB=2, 5
CLASSCFG[DEFAULT] MAXNODE=16, 32
CLASSCFG[batch] MAXNODE=12, 32
QOSCFG[hiprio] MAXJOB=3, 5 MAXNODE=32, 64
```

**i** Job [preemption](#) status can be adjusted based on whether the job violates a soft policy using the [ENABLESPVIOLATIONPREEMPTION](#) parameter.

## Per-partition Limits

Per-partition scheduling can set limits and enforce credentials and policies on a per-partition basis. Configuration for per-partition scheduling is done on the grid head. In a grid, each Moab cluster is considered a partition. Per-partition scheduling is typically used in a Master/Slave grid.

To enable per-partition scheduling, add the following to `moab.cfg`:

```
PERPARTITIONSCHEDULING TRUE
JOBMIGRATEPOLICY JUSTINTIME
```

**i** With per-partition scheduling, it is recommended that limits go on the specific partitions and not on the global level. If limits are specified on both levels, Moab will take the more constricting of the limits. Also, please note that a `DEFAULT` policy on the global partition is not overridden by any policy on a specific partition.

## Per-partition Limits

You can configure per-job limits and credential usage limits on a per-partition basis in the `moab.cfg` file. Here is a sample configuration for partitions `g02` and `g03` in `moab.cfg`.

```
PARCFG[g02] CONFIGFILE=/opt/moab/parg02.cfg
PARCFG[g03] CONFIGFILE=/opt/moab/parg03.cfg
```

You can then add per-partition limits in each partition configuration file:

```
# /opt/moab/parg02.cfg
CLASSCFG[pbatch] MAXJOB=5
```

```
# /opt/moab/parg03.cfg
CLASSCFG[pbatch] MAXJOB=10
```

You can configure Moab so that jobs submitted to any partition besides *g02* and *g03* get the default limits in `moab.cfg`:

```
st1
CLASSCFG[pbatch] MAXJOB=2
```

## Supported Credentials and Limits

The user, group, account, QoS, and class credentials are supported in per-partition scheduling.

The following per-job limits are supported:

- [MAX.NODE](#)
- [MAX.WCLIMIT](#)
- [MAX.PROC](#)

The following credential usage limits are supported:

- [MAXJOB](#)
- [MAXNODE](#)
- [MAXPROC](#)
- [MAXWC](#)
- [MAXSUBMITJOBS](#)

[Multi-dimensional limits](#) are supported for the listed credentials and per-job limits. For example:

```
CLASSCFG[pbatch] MAXJOB[user:frank]=10
```

## Usage-based limits

Resource usage limits constrain the amount of resources a given job may consume. These limits are generally proportional to the resources requested and may include walltime, any standard resource, or any specified generic resource. The parameter [RESOURCELIMITPOLICY](#) controls which resources are limited, what limit policy is enforced per resource, and what actions the scheduler should take in the event of a policy violation.

## Configuring Actions

The [RESOURCELIMITPOLICY](#) parameter accepts a number of policies, resources, and actions using the format and values defined below.

**i** If walltime is the resource to be limited, be sure that the resource manager is configured to not interfere if a job surpasses its given walltime. For TORQUE, this is done by using [\\$ignwalltime](#) in the configuration on each MOM node.

## Format

RESOURCELIMITPOLICY<RESOURCE>: [ <SPOLICY>, ] <HPOLICY>: [ <SACTION>, ] <HACTION> [ : [ <SVIOLATIONTIME>, ] <HVIOLATIONTIME> ] . . .

Resource	Description
<b>CPUTIME</b>	Maximum total job proc-seconds used by any single job (allows scheduler enforcement of cpulimit).
<b>DISK</b>	Local disk space (in MB) used by any single job task.
<b>JOBMEM</b>	Maximum real memory/RAM (in MB) used by any single job. <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 5px;">  JOBMEM will only work with the <a href="#">MAXMEM</a> flag.         </div>
<b>JOBPROC</b>	Maximum processor load associated with any single job. You must set <a href="#">MAXPROC on page 417</a> to use <b>JOBPROC</b> .
<b>MEM</b>	Maximum real memory/RAM (in MB) used by any single job task.
<b>MINJOBPROC</b>	Minimum processor load associated with any single job (action taken if job is using 5% or less of potential CPU usage).
<b>NETWORK</b>	Maximum network load associated with any single job task.
<b>PROC</b>	Maximum processor load associated with any single job task.
<b>SWAP</b>	Maximum virtual memory/SWAP (in MB) used by any single job task.
<b>WALLTIME</b>	Requested job walltime.

Policy	Description
<b>ALWAYS</b>	take action whenever a violation is detected
<b>EXTENDEDVIOLATION</b>	take action only if a violation is detected and persists for greater than the specified time limit
<b>BLOCKEDWORKLOADONLY</b>	take action only if a violation is detected and the constrained resource is required by another job

Action	Description
<b>CANCEL</b>	terminate the job
<b>CHECKPOINT</b>	<a href="#">checkpoint</a> and terminate job
<b>MIGRATE</b>	requeue the job and require a different set of hosts for execution
<b>NOTIFY</b>	notify admins and job owner regarding violation
<b>REQUEUE</b>	terminate and requeue the job
<b>SUSPEND</b>	suspend the job and leave it suspended for an amount of time defined by the <a href="#">MINADMINSTIME</a> parameter

*Example 3-94: Notify and then cancel job if requested memory is exceeded*

```
# if job exceeds memory usage, immediately notify owner
# if job exceeds memory usage for more than 5 minutes, cancel the job
RESOURCELIMITPOLICY MEM:ALWAYS,EXTENDEDVIOLATION:NOTIFY,CANCEL:00:05:00
```

*Example 3-95: Checkpoint job on walltime violations*

```
# if job exceeds requested walltime, checkpoint job
RESOURCELIMITPOLICY WALLTIME:ALWAYS:CHECKPOINT
# when checkpointing, send term signal, followed by kill 1 minute later
RMCFG[base] TYPE=PBS CHECKPOINTTIMEOUT=00:01:00 CHECKPOINTSIG=SIGTERM
```

*Example 3-96: Cancel jobs that use 5% or less of potential CPU usage for more than 5 minutes*

```
RESOURCELIMITPOLICY MINJOBPROC:EXTENDEDVIOLATION:CANCEL:5:00
```

*Example 3-97: Migrating a job when it blocks other workload*

```
RESOURCELIMITPOLICY JOBPROC:BLOCKEDWORKLOADONLY:MIGRATE
```

## Specifying Hard and Soft Policy Violations

Moab is able to perform different actions for both hard and soft policy violations. In most resource management systems, a mechanism does not exist to allow the user to specify both hard and soft limits. To address this, Moab provides the [RESOURCELIMITMULTIPLIER](#) parameter that allows per partition and per resource multiplier factors to be specified to generate the actual hard and soft limits to be used. If the factor is less than one, the soft limit will be lower than the specified value and a Moab action will be taken before the specified limit is reached. If the factor is greater than one, the hard limit will be set higher than the specified limit allowing a buffer space before the hard limit action is taken.

In the following example, job owners will be notified by email when their memory reaches 100% of the target, and the job will be canceled if it reaches 125% of the target. For wallclock usage, the job will be

requeued when it reaches 90% of the specified limit if another job is waiting for its resources, and it will be checkpointed when it reaches the full limit.

```
RESOURCELIMITPOLICY      MEM:ALWAYS, ALWAYS:NOTIFY, CANCEL
RESOURCELIMITPOLICY      WALLTIME:BLOCKEDWORKLOADONLY, ALWAYS:REQUEUE, CHECKPOINT
RESOURCELIMITMULTIPLIER  MEM:1.25, WALLTIME:0.9
```

## Constraining Walltime Usage

While Moab constrains walltime using the parameter [RESOURCELIMITPOLICY](#) like other resources, it also allows walltime exception policies which are not available with other resources. In particular, Moab allows jobs to exceed the requested wallclock limit by an amount specified on a global basis using the [JOBMAXOVERRUN](#) parameter or on a per credential basis using the **OVERRUN** attribute of the [CLASSCFG](#) parameter.

```
JOBMAXOVERRUN      00:10:00
CLASSCFG [debug]   overrun=00:00:30
```

### Related topics

- [RESOURCELIMITPOLICY](#) parameter
- [FSTREE](#) parameter (set usage limits within share tree hierarchy)
- [Credential Overview](#)
- [JOBMAXOVERRUN](#) parameter
- [WCVIOLATIONACTION](#) parameter
- [RESOURCELIMITMULTIPLIER](#) parameter

## Fairshare

Fairshare allows historical resource utilization information to be incorporated into job feasibility and priority decisions. This feature allows site administrators to set system utilization targets for users, groups, accounts, classes, and QoS levels. Administrators can also specify the time frame over which resource utilization is evaluated in determining whether the goal is being reached. Parameters allow sites to specify the utilization metric, how historical information is aggregated, and the effect of fairshare state on scheduling behavior. You can specify fairshare targets for any credentials (such as user, group, and class) that administrators want such information to affect.

- [Fairshare Parameters](#)
  - [FSPOLICY - Specifying the Metric of Consumption](#)
  - [Specifying Fairshare Timeframe](#)
  - [Managing Fairshare Data](#)
- [Using Fairshare Information](#)
  - [Fairshare Targets](#)
  - [Fairshare Caps](#)

- [Priority-Based Fairshare](#)
- [Per-Credential Fairshare Weights](#)
- [Extended Fairshare Examples](#)
- [Hierarchical Fairshare/Share Trees](#)
  - [Defining the Tree](#)
  - [Controlling Tree Evaluation](#)
- [Importing Fairshare Data](#)

## Fairshare Parameters

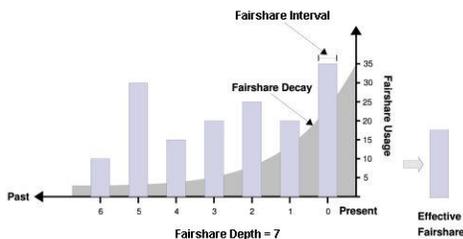
Fairshare is configured at two levels. First, at a system level, configuration is required to determine how fairshare usage information is to be collected and processed. Second, some configuration is required at the credential level to determine how this fairshare information affects particular jobs. The following are system level parameters:

Parameter	Description
<a href="#">FSINTERVAL</a>	Duration of each fairshare window.
<a href="#">FSDEPTH</a>	Number of fairshare windows factored into current fairshare utilization.
<a href="#">FSDECAY</a>	Decay factor applied to weighting the contribution of each fairshare window.
<a href="#">FSPOLICY</a>	Metric to use when tracking fairshare usage.

Credential level configuration consists of specifying fairshare utilization targets using the **\*CFG** suite of parameters, including [ACCOUNTCFG](#), [CLASSCFG](#), [GROUPCFG](#), [QOSCFG](#), and [USERCFG](#).

If global (multi-cluster) fairshare is used, Moab must be configured to synchronize this information with an [identity manager](#).

Image 3-3: Effective fairshare over 7 days



## FSPOLICY - Specifying the Metric of Consumption

As Moab runs, it records how available resources are used. Each iteration (**RMPOLLINTERVAL** seconds) it updates fairshare resource utilization statistics. Resource utilization is tracked in accordance with the **FSPOLICY** parameter allowing various aspects of resource consumption information to be measured. This parameter allows selection of both the types of resources to be tracked as well as the method of tracking. It provides the option of tracking usage by dedicated or consumed resources, where dedicated usage tracks what the scheduler assigns to the job and consumed usage tracks what the job actually uses.

Metric	Description
<b>DEDICATEDPES</b>	Usage tracked by <a href="#">processor-equivalent</a> seconds dedicated to each job. This is based on the total number of dedicated processor-equivalent seconds delivered in the system. Useful in dedicated and shared nodes environments.
<b>DEDICATEDPS</b>	Usage tracked by processor seconds dedicated to each job. This is based on the total number of dedicated processor seconds delivered in the system. Useful in dedicated node environments.
<b>DEDICATEDPS%</b>	Usage tracked by processor seconds dedicated to each job. This is based on the total number of dedicated processor seconds <i>available</i> in the system.
<b>UTILIZEDPS</b>	Usage tracked by processor seconds used by each job. This is based on the total number of utilized processor seconds delivered in the system. Useful in shared node/SMP environments.

Example 3-98:

An example may clarify the use of the **FSPOLICY** parameter. Assume a 4-processor job is running a parallel `/bin/sleep` for 15 minutes. It will have a dedicated fairshare usage of 1 processor-hour but a consumed fairshare usage of essentially nothing since it did not consume anything. Most often, dedicated fairshare usage is used on dedicated resource platforms while consumed tracking is used in shared SMP environments.

```
FSPOLICY      DEDICATEDPS%
FSINTERVAL    24:00:00
FSDEPTH       28
FSDECAY       0.75
```

## Specifying Fairshare Timeframe

When configuring fairshare, it is important to determine the proper timeframe that should be considered. Many sites choose to incorporate historical usage information from the last one to two weeks while others are only concerned about the events of the last few hours. The correct setting is very site dependent and usually incorporates both average job turnaround time and site mission policies.

With Moab's fairshare system, time is broken into a number of distinct fairshare windows. Sites configure the amount of time they want to consider by specifying two parameters, **FSINTERVAL** and **FSDEPTH**. The **FSINTERVAL** parameter specifies the duration of each window while the **FSDEPTH**

parameter indicates the number of windows to consider. Thus, the total time evaluated by fairshare is simply **FSINTERVAL \* FSDEPTH**.

Many sites want to limit the impact of fairshare data according to its age. The **FSDECAY** parameter allows this, causing the most recent fairshare data to contribute more to a credential's total fairshare usage than older data. This parameter is specified as a standard decay factor, which is applied to the fairshare data. Generally, decay factors are specified as a value between 1 and 0 where a value of 1 (the default) indicates no decay should be specified. The smaller the number, the more rapid the decay using the calculation  $\text{WeightedValue} = \text{Value} * \langle \text{DECAY} \rangle ^ \langle N \rangle$  where  $\langle N \rangle$  is the window number. The following table shows the impact of a number of commonly used decay factors on the percentage contribution of each fairshare window.

Decay Factor	Wino	Win1	Win2	Win3	Win4	Win5	Win6	Win7
1.00	100%	100%	100%	100%	100%	100%	100%	100%
0.80	100%	80%	64%	51%	41%	33%	26%	21%
0.75	100%	75%	56%	42%	31%	23%	17%	12%
0.50	100%	50%	25%	13%	6%	3%	2%	1%

While selecting how the total fairshare time frame is broken up between the number and length of windows is a matter of preference, it is important to note that more windows will cause the decay factor to degrade the contribution of aged data more quickly.

## Managing Fairshare Data

Using the selected fairshare usage metric, Moab continues to update the current fairshare window until it reaches a fairshare window boundary, at which point it rolls the fairshare window and begins updating the new window. The information for each window is stored in its own file located in the Moab statistics directory. Each file is named `FS.<EPOCHTIME>[.<PNAME>]` where `<EPOCHTIME>` is the time the new fairshare window became active (see [sample data file](#)) and `<PNAME>` is only used if per-partition [share trees](#) are configured. Each window contains utilization information for each entity as well as for total usage.

**i** Historical fairshare data is recorded in the fairshare file using the metric specified by the **FSPOLICY** parameter. By default, this metric is processor-seconds.

**i** Historical fairshare data can be directly analyzed and reported using the `midag -f -v` command.

When Moab needs to determine current fairshare usage for a particular credential, it calculates a decay-weighted average of the usage information for that credential using the most recent fairshare intervals where the number of windows evaluated is controlled by the **FSDEPTH** parameter. For example, assume the credential of interest is user *john* and the following parameters are set:

```

FSINTERVAL 12:00:00
FSDEPTH    4
FSDECAY    0.5
    
```

Further assume that the fairshare usage intervals have the following usage amounts:

Fairshare interval	Total user <i>john</i> usage	Total cluster usage
0	60	110
1	0	125
2	10	100
3	50	150

Based on this information, the current fairshare usage for user *john* would be calculated as follows:

$$Usage = (60 * 1 + .5^1 * 0 + .5^2 * 10 + .5^3 * 50) / (110 + .5^1*125 + .5^2*100 + .5^3*150)$$

**i** The current fairshare usage is relative to the actual resources delivered by the system over the timeframe evaluated, not the resources available or configured during that time.

**i** Historical fairshare data is organized into a number of data files, each file containing the information for a length of time as specified by the `FSINTERVAL` parameter. Although `FSDEPTH`, `FSINTERVAL`, and `FSDECAY` can be freely and dynamically modified, such changes may result in unexpected fairshare status for a period of time as the fairshare data files with the old `FSINTERVAL` setting are rolled out.

## Using Fairshare Information

### Fairshare Targets

Once the global fairshare policies have been configured, the next step involves applying resulting fairshare usage information to affect scheduling behavior. As mentioned in the Fairshare Overview, by specifying fairshare targets, site administrators can configure how fairshare information impacts scheduling behavior. The targets can be applied to user, group, account, QoS, or class credentials using the `FSTARGET` attribute of `*CFG` credential parameters. These targets allow fairshare information to affect job priority and each target can be independently selected to be one of the types documented in the following table:

Target type - Ceiling	
Target modifier	-

Target type - Ceiling	
<b>Job impact</b>	Priority
<b>Format</b>	Percentage Usage
<b>Description</b>	Adjusts job priority down when usage exceeds target. See <a href="#">How violated ceilings and floors affect fairshare-based priority on page 383</a> for more information on how ceilings affect job priority.

Target type - Floor	
<b>Target modifier</b>	+
<b>Job impact</b>	Priority
<b>Format</b>	Percentage Usage
<b>Description</b>	Adjusts job priority up when usage falls below target. See <a href="#">How violated ceilings and floors affect fairshare-based priority on page 383</a> for more information on how floors affect job priority.

Target type - Target	
<b>Target modifier</b>	N/A
<b>Job impact</b>	Priority
<b>Format</b>	Percentage Usage
<b>Description</b>	Adjusts job priority when usage does not meet target.

**i** Setting a fairshare target value of *0* indicates that there is no target and that the priority of jobs associated with that credential should not be affected by the credential's previous fairshare target. If you want a credential's cluster usage near 0%, set the target to a very small value, such as *0.001*.

**Example**

The following example increases the priority of jobs belonging to user *john* until he reaches 16.5% of total cluster usage. All other users have priority adjusted both up and down to bring them to their target usage of 10%:

```

FSPOLICY          DEDICATEDPS
FSWEIGHT          1
FSUSERWEIGHT     100
USERCFG[john]    FSTARGET=16.5+
USERCFG[DEFAULT] FSTARGET=10
...
    
```

### Fairshare Caps

Where fairshare targets affect a job's priority and position in the eligible queue, fairshare caps affect a job's eligibility. Caps can be applied to users, accounts, groups, classes, and QoSs using the **FSCAP** attribute of **\*CFG** credential parameters and can be configured to modify scheduling behavior. Unlike fairshare targets, if a credential reaches its fairshare cap, its jobs can no longer run and are thus removed from the eligible queue and placed in the blocked queue. In this respect, fairshare targets behave like soft limits and fairshare caps behave like hard limits. Fairshare caps can be absolute or relative as described in the following table. If no modifier is specified, the cap is interpreted as relative.

Absolute Cap	
<b>Cap Modifier:</b>	^
<b>Job Impact:</b>	Feasibility
<b>Format:</b>	Absolute Usage
<b>Description:</b>	Constrains job eligibility as an absolute quantity measured according to the scheduler charge metric as defined by the <a href="#">FSPOLICY</a> parameter

Relative Cap	
<b>Cap Modifier:</b>	%
<b>Job Impact:</b>	Feasibility
<b>Format:</b>	Percentage Usage
<b>Description:</b>	Constrains job eligibility as a percentage of total delivered cycles measured according to the scheduler charge metric as defined by the <a href="#">FSPOLICY</a> parameter.

### Example

The following example constrains the *marketing* account to use no more than **16,500** processor seconds during any given floating one week window. At the same time, all other accounts are constrained to use no more than **10%** of the total delivered processor seconds during any given one week window.

```
FSPOLICY          DEDICATEDPS
FSINTERVAL        12:00:00
FSDEPTH           14
ACCOUNTCFG[marketing] FSCAP=16500^
ACCOUNTCFG[DEFAULT]  FSCAP=10
...
```

## Priority-Based Fairshare

The most commonly used type of fairshare is priority based fairshare. In this mode, fairshare information does not affect whether a job can run, but rather only the job's priority relative to other jobs. In most cases, this is the desired behavior. Using the standard fairshare target, the priority of jobs of a particular user who has used too many resources over the specified fairshare window is lowered. Also, the standard fairshare target increases the priority of jobs that have not received enough resources.

While the standard fairshare target is the most commonly used, Moab can also specify fairshare ceilings and floors. These targets are like the default target; however, ceilings only adjust priority down when usage is too high and floors only adjust priority up when usage is too low.

Since fairshare usage information must be integrated with Moab's overall priority mechanism, it is critical that the corresponding fairshare priority weights be set. Specifically, the [FSWEIGHT](#) component weight parameter and the target type subcomponent weight (such as [FSACCOUNTWEIGHT](#), [FSCCLASSWEIGHT](#), [FSGROUPWEIGHT](#), [FSQOSWEIGHT](#), and [FSUSERWEIGHT](#)) be specified.



If these weights are not set, the fairshare mechanism will be enabled but have no effect on scheduling behavior. See the [Job Priority Factor Overview](#) for more information on setting priority weights.

## Example

```
# set relative component weighting
FSWEIGHT          1
FSUSERWEIGHT      10
FSGROUPWEIGHT     50

FSINTERVAL        12:00:00
FSDEPTH           4
FSDECAY           0.5
FSPOLICY          DEDICATEDPS
# all users should have a FS target of 10%
USERCFG[DEFAULT]  FSTARGET=10.0
# user john gets extra cycles
USERCFG[john]     FSTARGET=20.0
# reduce staff priority if group usage exceed 15%
GROUPCFG[staff]   FSTARGET=15.0-
# give group orion additional priority if usage drops below 25.7%
GROUPCFG[orion]   FSTARGET=25.7+
```

**i** Job preemption status can be adjusted based on whether the job violates a fairshare target using the `ENABLEFSVIOLATIONPREEMPTION` parameter.

## Credential-Specific Fairshare Weights

Credential-specific fairshare weights can be set using the **FSWEIGHT** attribute of the `ACCOUNT`, `GROUP`, and `QOS` credentials as in the following example:

```
FSWEIGHT 1000
ACCOUNTCFG[orion1] FSWEIGHT=100
ACCOUNTCFG[orion2] FSWEIGHT=200
ACCOUNTCFG[orion3] FSWEIGHT=-100
GROUPCFG[staff] FSWEIGHT=10
```

If specified, a per-credential fairshare weight is added to the global component fairshare weight.

**i** The **FSWEIGHT** attribute is only enabled for `ACCOUNT`, `GROUP`, and `QOS` credentials.

## Extended Fairshare Examples

### Example 3-99: Multi-Cred Cycle Distribution

Example 1 represents a university setting where different schools have access to a cluster. The Engineering department has put the most money into the cluster and therefore has greater access to the cluster. The Math, Computer Science, and Physics departments have also pooled their money into the cluster and have reduced relative access. A support group also has access to the cluster, but since they only require minimal compute time and shouldn't block the higher-paying departments, they are constrained to five percent of the cluster. At this time, users Tom and John have specific high-priority projects that need increased cycles.

```

#global general usage limits - negative priority jobs are considered in scheduling
ENABLENEGJOBPRIORITY TRUE
# site policy - no job can last longer than 8 hours
USERCFG[DEFAULT] MAX.WCLIMIT=8:00:00
# Note: default user FS target only specified to apply default user-to-user balance
USERCFG[DEFAULT] FSTARGET=1
# high-level fairshare config
FSPOLICY DEDICATEDPS
FSINTERVAL 12:00:00
FSDEPTH 32 #recycle FS every 16 days
FSDECAY 0.8 #favor more recent usage info
# qos config
QOSCFG[inst] FSTARGET=25
QOSCFG[supp] FSTARGET=5
QOSCFG[premium] FSTARGET=70
# account config (QoS access and fstargets)
# Note: user-to-account mapping handled via allocation manager
# Note: FS targets are percentage of total cluster, not percentage of QOS
ACCOUNTCFG[cs] QLIST=inst FSTARGET=10
ACCOUNTCFG[math] QLIST=inst FSTARGET=15

ACCOUNTCFG[phys] QLIST=supp FSTARGET=5
ACCOUNTCFG[eng] QLIST=premium FSTARGET=70
# handle per-user priority exceptions
USERCFG[tom] PRIORITY=100
USERCFG[john] PRIORITY=35
# define overall job priority
USERWEIGHT 10 # user exceptions
# relative FS weights (Note: QOS overrides ACCOUNT which overrides USER)
FSUSERWEIGHT 1
FSACCOUNTWEIGHT 10
FSQOSWEIGHT 100
# apply XFactor to balance cycle delivery by job size fairly
# Note: queuetime factor also on by default (use QUEUETIMEWEIGHT to adjust)
XFACTORWEIGHT 100
# enable preemption
PREEMPTPOLICY QUEUEUE
# temporarily allow phys to preempt math
ACCOUNTCFG[phys] JOBFLAGS=PREEMPTOR PRIORITY=1000
ACCOUNTCFG[math] JOBFLAGS=PREEMPTTEE

```

## Hierarchical Fairshare/Share Trees

Moab supports arbitrary depth hierarchical fairshare based on a share tree. In this model, users, groups, classes, and accounts can be arbitrarily organized and their usage tracked and limited. Moab extends common share tree concepts to allow mixing of credential types, enforcement of ceiling and floor style usage targets, and mixing of hierarchical fairshare state with other priority components.

### Defining the Tree

The `FSTREE` parameter can be used to define and configure the share tree used in fairshare configuration. This parameter supports the following attributes:

SHARES	
<b>Format:</b>	<COUNT>[@<PARTITION>][,<COUNT>[@<PARTITION>]]... where <COUNT> is a double and <PARTITION> is a specified partition name.
<b>Description:</b>	Specifies the node target usage or share.
<b>Example:</b>	<pre>FSTREE [Eng]    SHARES=1500.5 FSTREE [Sales] SHARES=2800</pre>

MEMBERLIST	
<b>Format:</b>	Comma delimited list of child nodes of the format [<OBJECT_TYPE>]:<OBJECT_ID> where object types are only specified for <i>leaf nodes</i> associated with <b>user</b> , <b>group</b> , <b>class</b> , <b>qos</b> , or <b>acct</b> credentials.
<b>Description:</b>	Specifies the tree objects associated with this node.
<b>Example:</b>	<pre>FSTREE [root]    SHARES=100    MEMBERLIST=Eng, Sales FSTREE [Eng]    SHARES=1500.5  MEMBERLIST=user:john,user:steve,user:bob FSTREE [Sales]  SHARES=2800    MEMBERLIST=Sales1, Sales2, Sales3 FSTREE [Sales1] SHARES=30      MEMBERLIST=user:kellyp,user:sam FSTREE [Sales2] SHARES=10      MEMBERLIST=user:ux43,user:ux44,user:ux45 FSTREE [Sales3] SHARES=60      MEMBERLIST=user:robert,user:tjackson</pre>

Current tree configuration and monitored usage distribution is available using the [mdiag -f -v](#) commands.

### Controlling Tree Evaluation

Moab provides multiple policies to customize how the share tree is evaluated.

Policy	Description
<b>FSTREETIERMULTIPLIER</b>	Decreases the value of sub-level usage discrepancies. It can be a positive or negative value. When positive, the parent's usage in the tree takes precedence; when negative, the child's usage takes precedence. The usage amount is not changed, only the coefficient used when calculating the value of fstree usage in priority. When using this parameter, it is recommended that you research how it changes the values in <code>mdiag -p</code> to determine the appropriate use.
<b>FSTREECAP</b>	Caps lower level usage factors to prevent them from exceeding upper tier discrepancies.

### Using FS Floors and Ceilings with Hierarchical Fairshare

All standard fairshare facilities including target floors, target ceilings, and target caps are supported when using hierarchical fairshare.

### Multi-Partition Fairshare

Moab supports independent, per-partition hierarchical fairshare targets allowing each partition to possess independent prioritization and usage constraint settings. This is accomplished by setting the **PERPARTITIONSCHEDULING** attribute of the **FSTREE** parameter to **TRUE** in `moab.cfg` and setting `partition="name"` in your `<fstree>` leaf.

```
FSTREE [tree]
<fstree>
  <tnode partition="slave1" name="root" type="acct" share="100" limits="MAXJOB=6">
    <tnode name="accta" type="acct" share="50" limits="MAXSUBMITJOBS=2 MAXJOB=1">
      <tnode name="fred" type="user" share="1" limits="MAXWC=1:00:00">
      </tnode>
    </tnode>
    <tnode name="acctb" type="acct" share="50" limits="MAXSUBMITJOBS=4 MAXJOB=3">
      <tnode name="george" type="user" share="1" >
      </tnode>
    </tnode>
  </tnode>
  <tnode partition="slave2" name="root" type="acct" share="100"
limits="MAXSUBMITJOBS=6 MAXJOB=5">
    <tnode name="accta" type="acct" share="50">
      <tnode name="paul" type="user" share="1">
      </tnode>
    </tnode>
    <tnode name="acctb" type="acct" share="50">
      <tnode name="ringo" type="user" share="1">
      </tnode>
    </tnode>
  </tnode>
</fstree>
```

**i** If no partition is specified for a given share value, then this value is assigned to the global partition. If a partition exists for which there are no explicitly specified shares for any node, this partition will use the share distribution assigned to the global partition.

### Dynamically Importing Share Tree Data

Share trees can be centrally defined within a database, flat file, information service, or other system and this information can be dynamically imported and used within Moab by setting the **FSTREE** parameter within the [Identity Managers on page 786](#). This interface can be used to load current information at startup and periodically synchronize this information with the master source.

### To create a fairshare tree in a separate XML file and import it into Moab

1. Create a file to store your fair share tree specification. Give it a descriptive name and store it in your Moab home directory (`$MOABHOMEDIR` or `$MOABHOMEDIR/etc`). In this example, the file is called `fstree.dat`.
2. In the first line of `fstree.dat`, set **FSTREE**[myTree] to indicate that this is a fairshare file.
3. Build a tree in XML to match your needs. For example:

```
FSTREE [myTree]
<fstree>
<tnode name="root" share="100">
<tnode name="john" type="user" share="50" limits="MAXJOB=8 MAXPROC=24
MAXWC=01:00:00"></tnode>
<tnode name="jane" type="user" share="50" limits="MAXJOB=5"></tnode>
</tnode>
</fstree>
```

*This configuration creates a fairshare tree in which users share a value of 100. Users john and jane share the value equally, because each has been given 50.*

Because 100 is an arbitrary number, users john and jane could be assigned 10000 and 10000 respectively and still have a 50% share under the parent leaf. To keep the example simple, however, it is recommended that you use 100 as your arbitrary share value and distribute the share as percentages. In this case, john and jane each have 50%.

If the users' numbers do not add up to at least the fairshare value of 100, the remaining value is shared among all users under the tree. For instance, if the tree had a value of 100, user john had a value of 50, and user jane had a value of 25, then 25% of the fairshare tree value would belong to all other users associated with the tree. By default, tree leaves do not limit who can run under them.

**i** Each value specified in the `tnode` elements must be contained in quotation marks.

- Optional: Share trees defined within a flat file can be cumbersome; consider running `tidy` for xml to improve readability. Sample usage:

```
> tidy -i -xml goldy.cfg <filename> <output file>

# Sample output

FSTREE [myTree]
<fstree>
  <tnode name="root" share="100">
    <tnode name="john" type="user" share="50" limits="MAXJOB=8
MAXPROC=24 MAXWC=01:00:00">
    </tnode>
    <tnode name="jane" type="user" share="50" limits="MAXJOB=5">
    </tnode>
  </tnode>
</fstree>
```

- Link the new file to Moab using the `IDCFG` parameter in your Moab configuration file.

```
IDCFG [myTree] server="FILE:/// $MOABH OMEDIR/etc/fstree.dat" REFRESHPERIOD=INFINITY
```

*Moab imports the myTree fairshare tree from the fstree.dat file. Setting `REFRESHPERIOD` to `INFINITY` causes Moab to read the file each time it starts or restarts, but other settings (hour, day, month) cause Moab to read the file more often (See [Refreshing Identity Manager Data](#) for more information).*

- To view your fairshare tree configuration, run `mdiag -f`. If it is configured correctly, the tree information will appear beneath all the information about your fairshare settings configured in `moab.cfg`.

```

> mdiag -f
Share Tree Overview for partition 'ALL'
Name          Usage      Target      (FSFACTOR)
-----
root          100.00    100.00    of  100.00 (node: 1171.81) (0.00)
- john        16.44     50.00    of  100.00 (user: 192.65) (302.04) MAXJOB=8
MAXPROC=24 MAXWC=3600
- jane        83.56     50.00    of  100.00 (user: 979.16) (-302.04) MAXJOB=5

```

*The settings you configured in `fstree.dat` appear in the output. The tree of 100 is shared equally between users `john` and `jane`.*

### Specifying Share Tree Based Limits

Limits can be specified on internal nodes of the share tree using standard [credential limit semantics](#). The following credential usage limits are valid:

- **MAXJOB** (Maximum number of idle jobs allowed for the credential)
- [MAXJOB on page 416](#)
- [MAXMEM on page 416](#)
- [MAXNODE on page 416](#)
- [MAXPROC on page 417](#)
- [MAXSUBMITJOBS on page 418](#)
- [MAXWC on page 418](#)

#### Example 3-100: FSTREE limits example

```

FSTREE [myTree]
<fstree>
  <tnode name="root" share="100">
    <tnode name="john" type="user" share="50" limits="MAXJOB=8
      MAXPROC=24 MAXWC=01:00:00">
    </tnode>
    <tnode name="jane" type="user" share="50" limits="MAXJOB=5">
    </tnode>
  </tnode>
</fstree>

```

### Other Uses of Share Trees

If a share tree is defined, it can be used for purposes beyond fairshare, including organizing general usage and performance statistics for reporting purposes (see [showstats -T](#)), enforcement of tree node based usage limits, and specification of resource access policies.

#### Related topics

- [mdiag -f](#) command (provides diagnosis and monitoring of the fairshare facility)
- [FSENABLECAPRIORITY](#) parameter
- [ENABLEFSPREEMPTION](#) parameter
- [FSTARGETISABSOLUTE](#) parameter

## Sample FairShare Data File

```
FS.<EPOCHTIME>
```

```
# FS Data File (Duration: 43200 seconds) Starting: Sat Jul 8 06:00:20
user          jvella      134087.910
user          reynolds    98283.840
user          gastor      18751.770
user          uannan      145551.260
user          mwillis     149279.140
...
group         DEFAULT     411628.980
group         RedRock     3121560.280
group         Summit      500327.640
group         Arches      3047918.940
acct          Administration 653559.290
acct          Engineering 4746858.620
acct          Shared      75033.020
acct          Research    1605984.910
qos           Deadline    2727971.100
qos           HighPriority 4278431.720
qos           STANDARD    75033.020
class         batch       7081435.840
sched         iCluster    7081435.840
```

*The total usage consumed in this time interval is 7081435.840 processor-seconds. Since every job in this example scenario had a user, group, account, and QOS assigned to it, the sum of the usage of all members of each category should equal the total usage value:  $USERA + USERB + USERC + USERD = GROUPA + GROUPB = ACCTA + ACCTB + ACCTC = QOSO + QOS1 + QOS2 = SCHED$ .*

## Charging a Workflow

The first thing you need to do is install Moab Accounting Manager. You can do this through the [Adaptive Computing](#) website. After Moab Accounting Manager is installed you can look at the `createExampleGoldState.pl` script to see examples of how to define users, projects, accounts, and charge rates in Moab Accounting Manager. You can also run the script to set up a few test users, accounts, and charge rates in Moab Accounting Manager. If ever you want to clear the sqlite database in Moab Accounting Manager, run `./resetdb` as root with the first parameter being the accounting admin user and it will restore the accounting state the same as a fresh install.

## Add NAMI URLs to moab.cfg

The out-of-the-box-solution doesn't use the create or modify NAMI URL. Moab uses the reserve URL to reserve funds and charge for setup costs, the charge URL to reserve funds and charge for reoccurring costs, and the delete URL to unreserve the funds. *These three URLs must be defined in `moab.cfg` in order to charge a workflow.* You should set a Reserve Failure Action in `moab.cfg` (the bottom line in the example) in case the customer doesn't have enough funds to proceed. Therefore a sample `moab.cfg` might look like this in the NAMI section:

```
AMCFG[nami] TYPE=NATIVE
AMCFG[nami] StartURL=exec:///opt/moab/tools/mam/usage.start.mam.pl
AMCFG[nami] UpdateURL=exec:///opt/moab/tools/mam/usage.update.mam.pl
AMCFG[nami] PauseURL=exec:///opt/moab/tools/mam/usage.pause.mam.pl
AMCFG[nami] EndURL=exec:///opt/moab/tools/mam/usage.end.mam.pl
AMCFG[nami] DeleteURL=exec:///opt/moab/tools/mam/usage.delete.mam.pl
AMCFG[nami] FLUSHINTERVAL=hour
AMCFG[nami] StartFailureAction=HOLD
```

## How Workflows Are Expressed in nami.cfg

The `nami.cfg` file was designed to be similar to the `moab.cfg`. Each line is parsed as either **Predicate** [`constant`] **Attribute=***value* or **key** *value*. Each constant in the **JobTemplate** predicate identifies specific job templates in `moab.cfg`. In respect to the workflow, `nami.cfg` needs to know about:

- (required) What job template in the workflow is the infinite job that you will charge based on a recurring bill cycle. In order to specify this you must set the **Recurring** attribute to *True*. For example:

```
JobTemplate[newvm]                Recurring=True
```

- (required) What job template starts off the workflow where you will reserve funds for the eventual infinite job's first recurring bill as well as any setup costs that might occur. For example:

```
JobTemplate[newvm]                StartJob=newvmprovision
```

- (required) What job templates follow the infinite job. This is important to register these template names with NAMI in case of a job failure; otherwise, NAMI will simply ignore them.

```
JobTemplate[newvm]                TemplateDepend=newvmprovision
JobTemplate[newvm]                TemplateDepend=vmstoragehookup
```

- (optional) Moab Accounting Manager VBF charge rate associated with provision jobs. These charge rates will be fixed setup costs per provision job that will be summed up and reserved with the first recurring bill at the run time of the start job. They will be charged at run time of the infinite job to ensure that all provision jobs ran successfully. It is important to note that the run time of the infinite job is also when the bill cycle starts. The first recurring bill is not charged with the setup costs at the beginning of the first bill cycle but rather at the end of each bill cycle after the resources are used. The following is an example of configuring multiple set up costs:

```
JobTemplate[newvmprovision]       GoldChargeRateName=VmSetUpCost
JobTemplate[vmstoragehookup]     GoldChargeRateName=V
```

- (optional) In case you want to toggle billing on and off for a workflow, you can use the **TrackUsage** attribute. The attribute only needs to be set on the recurring job for each workflow. Since having billing on is the default behavior, here is an example of turning billing off:

```
JobTemplate[newvm]                TrackUsage=False
```

## Workflow from moab.cfg to nami.cfg Example

### moab.cfg

```

JOBCFG[newvmprovision]    GENERICSYSJOB=AType=exec,Action="$TOOLSDIR/newvmprovision.pl
$*.SM",EType=start,Timeout=2:00,Flags=objectxmlstdin
JOBCFG[newvmprovision]    INHERITRES=TRUE
JOBCFG[newvmprovision]    FLAGS=NORMSTART

JOBCFG[vmstoragehookup]
GENERICSYSJOB=AType=exec,Action="$TOOLSDIR/vmstoragehookup.pl $.SM
$.SMMAP",EType=start,Timeout=2:00,Flags=objectxmlstdin
JOBCFG[vmstoragehookup]  INHERITRES=TRUE
JOBCFG[vmstoragehookup]  FLAGS=NORMSTART
JOBCFG[vmstoragehookup]  TEMPLATEDEPEND=AFTEROK:newvmprovision

JOBCFG[newvm]             FLAGS=VMTRACKING    SELECT=true
JOBCFG[newvm]             TEMPLATEDEPEND=AFTEROK:vmstoragehookup
JOBCFG[newvm]             FLAGS=NORMSTART
JOBCFG[newvm]             DESTROYTEMPLATE=destroyvm
JOBCFG[newvm]             MIGRATETEMPLATE=migratevm

```

### nami.cfg

```

JobTemplate[newvm]        Recurring=True
JobTemplate[newvm]        StartJob=newvmprovision
JobTemplate[newvm]        TrackUsage=True
JobTemplate[newvm]        TemplateDepend=newvmprovision
JobTemplate[newvm]        TemplateDepend=vmstoragehookup
JobTemplate[newvmprovision] GoldChargeRateName=VmSetUpCost
JobTemplate[vmstoragehookup] GoldChargeRateName=VmStorageHookUpCost

```

## Understanding UsageAttributes in nami.cfg

Recurring jobs are represented in Moab Accounting Manager as usage records. A usage record helps constrain what account to charge, how much to charge, as well as keep record of additional information about the job. The `nami.cfg` file defines what the usage record will like for the recurring job. Each constant in the `UsageAttribute` can be an account constraint, charge rate name, usage record property name, or any combination of the three. Because the constant can be a usage record property (something that gets recorded and shown when you do a `glsusage`) and a charge rate name, most people make charge rate names match the usage record property. For example, *Processors* is something that you probably want to both record and charge for, so it is easier if the charge rate name and usage record attribute name are both called *Processors* and `nami.cfg` has the line:

```
UsageAttribute[Processors]    MoabXMLAttribute=Processors
```

The `nami.cfg` file interprets this to mean that the value in the *Processors* tag sent from Moab should be the same value passed in the *Processors* tag in the usage record sent to Moab Accounting Manager. Moab Accounting Manager will then recognize that it is the usage record property name, so it will record it and show it when you do a `glsusage` command and will also recognize that it is a charge rate so it will charge it accordingly. Other attributes like `user` and `project` help constrain which account gets charged. However, there is nothing preventing you from making additional charge rates in Moab Accounting Manager for them as well.

Currently, the NAMI scripts allow anything in the Moab job XML to be sent to Moab Accounting Manager, including gres and user-defined variables. Gres and variables are expressed with either a *gres:* or *var:* prepended to the name of what it is representing. For example you might see the following in `nami.cfg`:

```
UsageAttribute[Disk]           MoabXMLAttribute=GRes:Os
UsageAttribute[StorageHost]   MoabXMLAttribute=Var:SH
```

## Other Important nami.cfg Parameters

**SetupTime** - The amount of time (in seconds) beyond the bill cycle time that the reservation for the setup cost and first bill cycle should last before it destroys itself (defaults to 43200). Here is an example of how to define it in `nami.cfg`:

```
SetupTime                       3213 (Defaults to 43200)
```

**BillCycle** - The amount of time of the bill cycle. It helps determine how much funds to reserve for each bill cycle. Possible values are hour, day, week, month (the default is day). If you want to over- or under-reserve funds for the bill cycle, you can also configure the default values for hour (60), day (86400), week (604800), and month (2630000). Be careful to understand that if you use this with **RoundChargeDuration**, the charge duration will be rounded to the modified bill cycle which is probably not what you wanted to do. Configuring the bill cycle time is intended to increase or decrease the barrier of entry for your job and assumes that the charge duration is not rounded. Here is an example of setting the bill cycle and changing the time duration of *Month*.

```
BILLCYCLE                       Month      (Defaults to Day)
BILLCYCLE[Month]                 2640000 (Defaults to average seconds in a month)
```

**RoundChargeDuration** - If set to *True*, then the charge duration sent from Moab will be rounded up or down to match what was quoted within the default bounds of plus or minus 0 seconds. Otherwise, charge duration will not be rounded. The bound to round the charge duration is configurable with the **Bound** attribute. Here is an example of rounding the charge duration if it is within plus or minus a half a day:

```
RoundChargeDuration             True      (Defaults to False)
RoundChargeDuration[config]     Bound=43200 (Defaults to 0)
```

**Log** - If set to *True*, the `nami.log` will also include everything that is written to the `gold.log` in `/opt/mam/log/`. An example log configuration is:

```
LOG[config]      path=../log/nami.log      (Defaults to ../log/nami.log)
LOG[config]      loglevel=debug          (Possible values are fatal, error, warn, info,
debug and it defaults to info)
LOG[config]      includeGoldLog=false    (Defaults to true)
LOG[config]      permissions=640        (Defaults to 666)
LOG[config]      maxSize=10000000       (Defaults to 10000000)
LOG[config]      rolloverLimit=7        (Defaults to 7)
```

# Controlling Resource Access - Reservations, Partitions, and QoS Facilities

- [Advance Reservations](#) on page 450
- [Partitions](#) on page 494
- [Quality of Service \(QoS\) Facilities](#) on page 498

## Advance Reservations

An advance reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday. Advance reservation technology enables many features including [backfill](#), [deadline](#) based scheduling, [grid scheduling](#), and [QOS](#) support.

The [mrsvctl](#) command is used to [create](#), [modify](#), [query](#), and [release](#) reservations.

- [Reservation Overview](#)
- [Administrative Reservations](#)
- [Standing Reservations](#)
- [Reservation Policies](#)
- [Configuring and Managing Reservations](#)
- [Enabling Reservations for End-users](#)

## Reservation Overview

- [Resources](#)
- [TimeFrame](#)
- [Access Control List](#)
- [Job to Reservation Binding](#)
- [Reservation Specification](#)
- [Reservation Behavior](#)
- [Reservation Group](#)

Every reservation consists of 3 major components: (1) a set of resources, (2) a time frame, and (3) an access control list. Additionally, a reservation may also have a number of optional attributes controlling its behavior and interaction with other aspects of scheduling. Reservation attribute descriptions follow.

## Resources

Under Moab, the resources specified for a reservation are specified by way of a [task](#) description. Conceptually, a task can be thought of as an atomic, or indivisible, collection of resources. If reservation resources are unspecified, a task is a node by default. To define a task, specify resources. The resources may include processors, memory, swap, local disk, and so forth. For example, a single task may consist of one processor, 2 GB of memory, and 10 GB of local disk.

A reservation consists of one or more tasks. In attempting to locate the resources required for a particular reservation, Moab examines all feasible resources and locates the needed resources in groups specified by the task description. An example may help clarify this concept:

Reservation A requires four tasks. Each task is defined as 1 processor and 1 GB of memory.

Node X has 2 processors and 3 GB of memory available

Node Y has 2 processors and 1 GB of memory available

Node Z has 2 processors and 2 GB of memory available

When collecting the resources needed for the reservation, Moab examines each node in turn. Moab finds that Node X can support 2 of the 4 tasks needed by reserving 2 processors and 2 GB of memory, leaving 1 GB of memory unreserved. Analysis of Node Y shows that it can only support 1 task reserving 1 processor and 1 GB of memory, leaving 1 processor unreserved. Note that the unreserved memory on Node X cannot be combined with the unreserved processor on Node Y to satisfy the needs of another task because a task requires all resources to be located on the same node. Finally, analysis finds that node Z can support 2 tasks, fully reserving all of its resources.

Both reservations and jobs use the concept of a task description in specifying how resources should be allocated. It is important to note that although a task description is used to allocate resources to a reservation, this description does not in any way constrain the use of those resources by a job. In the above example, a job requesting resources simply sees 4 processors and 4 GB of memory available in reservation A. If the job has access to the reserved resources and the resources meet the other requirements of the job, the job could use these resources according to its own task description and needs.

Currently, the resources that can be associated with reservations include processors, memory, swap, local disk, initiator classes, and any number of arbitrary resources. Arbitrary resources may include peripherals such as tape drives, software licenses, or any other site specific resource.

## Time Frame

Associated with each reservation is a time frame. This specifies when the resources will be reserved or dedicated to jobs that meet the reservation's access control list (ACL). The time frame simply consists of a start time and an end time. When configuring a reservation, this information may be specified as a start time together with either an end time or a duration.

## Access Control List

A reservation's access control list specifies which jobs can use a reservation. Only jobs that meet one or more of a reservation's access criteria are allowed to use the reserved resources during the reservation time frame. Currently, the reservation access criteria include the following: users, groups, accounts, classes, QOS, job attributes, job duration, and job templates.

## Job to Reservation Binding

While a reservation's ACL will allow particular jobs to use reserved resources, it does not force any job to use these resources. With each job, Moab attempts to locate the best possible combination of available resources whether these are reserved or unreserved. For example, in the following figure, note that job X, which meets access criteria for both reservation A and B, allocates a portion of its resources from each reservation and the remainder from resources outside of both reservations.

Image 3-4: Job X uses resources from reservations A and B



Although by default, reservations make resources available to jobs that meet particular criteria, Moab can be configured to constrain jobs to only run within accessible reservations. This can be requested by the user on a job by job basis using a resource manager extension flag, or it can be enabled administratively via a QoS flag. For example, assume two reservations were created as follows:

```
> mrsvctl -c -a GROUP==staff -d 8:00:00 -h 'node[1-4]'
reservation staff.1 created
```

```
> mrsvctl -c -a USER==john -t 2
reservation john.2 created
```

If the user "john," who happened to also be a member of the group "staff," wanted to force a job to run within a particular reservation, "john" could do so using the **FLAGS** [resource manager extension](#).

Specifically, in the case of a PBS job, the following submission would force the job to run within the "staff.1" reservation.

```
> msub -l nodes=1,walltime=1:00:00,flags=ADVRES:staff.1 testjob.cmd
```

Note that for this to work, PBS needs to have resource manager extensions enabled as described in the [PBS Resource Manager Extension Overview](#). ([TORQUE](#) has resource manager extensions enabled by default.) If the user wants the job to run on reserved resources but does not care which, the user could submit the job with the following:

```
> msub -l nodes=1,walltime=1:00:00,flags=ADVRES testjob.cmd
```

To enable job to reservation mapping via [QoS](#), the QoS flag [USERRESERVED](#) should be set in a similar manner.

**i** Use the reservation [BYNAME](#) flag to require explicit binding for reservation access.

To lock jobs linked to a particular QoS into a reservation or reservation group, use the [REQRID](#) attribute.

## Reservation Specification

There are two main types of reservations that sites typically deal with. The first, administrative reservations, are typically one-time reservations created for special purposes and projects. These reservations are created using the [mrsvctl](#) or [setres](#) commands. These reservations provide an integrated mechanism to allow graceful management of unexpected system maintenance, temporary projects, and time critical demonstrations. This command allows an administrator to select a particular set of resources or just specify the quantity of resources needed. For example an administrator could use a regular expression to request a reservation be created on the nodes "blue0[1-9]" or could simply request that the reservation locate the needed resources by specifying a quantity based request such as "TASKS==20."

The second type of reservation is called a [standing reservation](#). It is specified using the [SRCFG](#) parameter and is of use when there is a recurring need for a particular type of resource distribution. Standing reservations are a powerful, flexible, and efficient means for enabling persistent or periodic policies such as those often enabled using [classes](#) or queues. For example, a site could use a standing reservation to reserve a subset of its compute resources for quick turnaround jobs during business hours on Monday thru Friday. The [Standing Reservation Overview](#) provides more information about configuring and using these reservations.

## Reservation Behavior

As previously mentioned, a given reservation may have one or more access criteria. A job can use the reserved resources if it meets at least one of these access criteria. It is possible to stack multiple reservations on the same node. In such a situation, a job can only use the given node if it has access to each active reservation on the node.

## Reservation Group

Reservations groups are ways of associating multiple reservations. This association is useful for [variable namespace](#) and [reservation requests](#). The reservations in a group inherit the variables from the reservation group head, but if the same variable is set locally on a reservation in the group, the local variable overrides the inherited variable. Variable inheritance is useful for [triggers](#) as it provides greater flexibility with automating certain tasks and system behaviors.

Jobs may be bound to a reservation group (instead of a single reservation) by using the resource manager extension [ADVRES](#).

## Infinite Jobs and Reservations

To allow infinite walltime jobs, you must have the following scheduler flag set:

```
SCHEDCFG[Moab] FLAGS=allowinfinitejobs
```

You can submit an infinite job by completing:

```
msub -l walltime=INFINITY
```

Or an infinite reservation by completing:

```
mrsvctl -c -d INFINITY
```

Infinite jobs can run in infinite reservations. Infinite walltime also works with job templates and advres.

Output XML for infinite jobs will print "INFINITY" in the ReqAWDDuration, and XML for infinite rsvs will print "INFINITY" in duration and endtime.

```
<Data>
  <rsv AUser="jgardner" AllocNodeCount="1" AllocNodeList="n5"
    AllocProcCount="4" AllocTaskCount="1" HostExp="n5"
    LastChargeTime="0" Name="jgardner.1" Partition="base"
    ReqNodeList="n5:1" Resources="PROCS=[ALL]" StatCAPS="0.00"
    StatCIPS="0.00" StatTAPS="0.00" StatTIPS="0.00" SubType="Other"
    Type="User" cost="0.000000" ctime="1302127058"
    duration="INFINITY" endtime="INFINITY" starttime="1302127058">
    <ACL aff="neutral" cmp="%" name="jgardner.1" type="RSV"></ACL>
    <ACL cmp="%" name="jgardner" type="USER"></ACL>
    <ACL cmp="%" name="company" type="GROUP"></ACL>
    <ACL aff="neutral" cmp="%" name="jgardner.1" type="RSV"></ACL>
    <History>
      <event state="PROCS=4" time="1302127058"></event>
    </History>
  </rsv>
</Data>
```

### Related topics

- [Reservation Allocation Policies](#)
- [Reservation Re-Allocation Policies](#)

## Administrative Reservations

- [Annotating Administrative Reservations](#)
- [Using Reservation Profiles](#)
- [Optimizing Maintenance Reservations](#)

Administrative reservations behave much like standing reservations but are generally created to address non-periodic, one-time issues. All administrative reservations are created using the `mrsvctl -c` (or `setres`) command and are persistent until they expire or are removed using the `mrsvctl -r` (or `releaseres`) command.

## Annotating Administrative Reservations

Reservations can be labeled and annotated using comments allowing other administrators, local users, portals and other services to obtain more detailed information regarding the reservations. Naming and annotations are configured using the `-n` and `-D` options of the `mrsvctl` command respectively, as in the following example:

```
> mrsvctl -c -D 'testing infiniband performance' -n nettest -h 'r:agt[15-245]'
```

## Using Reservation Profiles

You can set up reservation profiles to avoid manually and repetitively inputting standard reservation attributes. Profiles can specify reservation names, descriptions, ACLs, durations, hostlists, triggers, flags, and other aspects that are commonly used. With a reservation profile defined, a new administrative reservation can be created that uses this profile by specifying the `-P` flag as in the following example.

*Example 3-101:*

```
RSVPROFILE [mtn1] TRIGGER=Atype=exec,Action="/tmp/trigger1.sh",EType=start
RSVPROFILE [mtn1] USERLIST=steve,marym
RSVPROFILE [mtn1] HOSTEXP="r:50-250"
```

```
> mrsvctl -c -P mtn1 -s 12:00:00_10/03 -d 2:00:00
```

*Example 3-102: Non-Blocking System Reservations with Scheduler Pause*

```
RSVPROFILE [pause] TRIGGER=atype=exec,etype=start,action="/opt/moab/bin/mschedctl -p"
RSVPROFILE [pause] TRIGGER=atype=exec,etype=cancel,action="/opt/moab/bin/mschedctl -r"
RSVPROFILE [pause] TRIGGER=atype=exec,etype=end,action="/opt/moab/bin/mschedctl -r"
```

```
> mrsvctl -c -P pause -s 12:00:00_10/03 -d 2:00:00
```

## Optimizing Maintenance Reservations

Any reservation causes some negative impact on cluster performance as it further limits the scheduler's ability to optimize scheduling decisions. You can mitigate this impact by using flexible ACLs and triggers.

In particular, a maintenance reservation can be configured to reduce its effective reservation shadow by allowing overlap with checkpointable/preemptible jobs until the time the reservation becomes active. This can be done using a series of triggers that perform the following actions:

- Modify the reservation to disable preemption access.
- Preempt jobs that may overlap the reservation.
- Cancel any jobs that failed to properly checkpoint and exit.

The following example highlights one possible configuration:

```
RSVPROFILE[adm1] JOBATTRLIST=PREEMPTEE
RSVPROFILE[adm1] DESCRIPTION="regular system maintenance"
RSVPROFILE[adm1] TRIGGER=EType=start,Offset=-
300,AType=internal,Action="rsv:-:modify:acl:jattr-=PREEMPTEE"
RSVPROFILE[adm1] TRIGGER=EType=start,Offset=-240,AType=jobpreempt,Action="checkpoint"
RSVPROFILE[adm1] TRIGGER=EType=start,Offset=-60,AType=jobpreempt,Action="cancel"
```

```
> mrsvctl -c -P adm1 -s 12:00:00_10/03 -d 8:00:00 -h ALL
```

This reservation reserves all nodes in the cluster for a period of eight hours. Five minutes before the reservation starts, the reservation is modified to remove access to new preemptible jobs. Four minutes before the reservation starts, preemptible jobs that overlap the reservation are checkpointed. One minute before the reservation, all remaining jobs that overlap the reservation are canceled.

Reservations can also be used to evacuate virtual machines from a nodelist. To do this, you can configure a reservation profile in the `moab.cfg` file that calls an internal trigger to enable the evacuate VM logic. For example:

```
RSVPROFILE[evacvms]
TRIGGER=EType=start,AType=internal,action=node:$(HOSTLIST):evacvms
```

```
> mrsvctl -c -P evacvms -s 12:00:00_10/03 -d 8:00:00 -h ALL
```

Please note that Moab gives its best effort in evacuating VMs; however, if other reservations and policies prevent Moab from locating an alternate location for the VMs to be migrated to, then no action will occur. Administrators can attach additional triggers to the reservation profile to add evacuation logic where needed.

 You can also manually create a reservation that evacuates VMs from a nodelist by using the [EVACVMS](#) reservation flag. For example:

```
> mrsvctl -c -F EVACVMS -s 12:00:00_10/03 -d 8:00:00 -h ALL
```

## Related topics

- [Backfill](#)
- [Preemption](#)
- [mrsvctl](#) command

## Standing Reservations

Standing reservations build upon the capabilities of advance reservations to enable a site to enforce advanced usage policies in an efficient manner. Standing reservations provide a superset of the capabilities typically found in a batch queuing system's class or queue architecture. For example, queues can be used to allow only particular types of jobs access to certain compute resources. Also, some batch systems allow these queues to be configured so that they only allow this access during certain times of the day or week. Standing reservations allow these same capabilities but with greater flexibility and efficiency than is typically found in a normal queue management system.

Standing reservations provide a mechanism by which a site can dedicate a particular block of resources for a special use on a regular daily or weekly basis. For example, node X could be dedicated to running jobs only from users in the accounting group every Friday from 4 to 10 p.m. See the [Reservation Overview](#) for more information about the use of reservations. The [Managing Reservations](#) section provides a detailed explanation of the concepts and steps involved in the creation and configuration of standing reservations.

A standing reservation is a powerful means of doing the following:

- Controlling local credential based access to resources.
- Controlling external peer and grid based access to resources.
- Controlling job responsiveness and turnaround.

Related topics

- [SRCFG](#)
- [Moab Workload Manager for Grids](#)
- [mdiag -s](#) (diagnose standing reservations)

## Reservation Policies

- [Controlling Priority Reservation Creation](#)
- [Managing Resource Failures](#)
- [Resource Allocation Policy](#)
- [Resource Re-Allocation Policy](#)

## Controlling Priority Reservation Creation

In addition to standing and administrative reservations, Moab can also create priority reservations. These reservations are used to allow the benefits of out-of-order execution (such as is available with [backfill](#)) without the side effect of job starvation. Starvation can occur in any system where the potential exists for a job to be overlooked by the scheduler for an indefinite period. In the case of backfill, small jobs may continue to run on available resources as they become available while a large job sits in the queue, never able to find enough nodes available simultaneously on which to run.

To avoid such situations, priority reservations are created for high priority jobs that cannot run immediately. When making these reservations, the scheduler determines the earliest time the job could start and then reserves these resources for use by this job at that future time.

## Priority Reservation Creation Policy

Organizations have the ability to control how priority reservations are created and maintained. It is possible that one job can be at the top of the priority queue for a time and then get bypassed by another job submitted later. The parameter [RESERVATIONPOLICY](#) allows a site to determine how existing reservations should be handled when new reservations are made.

Value	Description
<b>HIGHEST</b>	<p>All jobs that have ever received a priority reservation up to the <b>RESERVATIONDEPTH</b> number will maintain that reservation until they run, even if other jobs later bypass them in priority value.</p> <p>For example, if there are four jobs with priorities of 8, 10,12, and 20.</p> <div style="border: 1px dashed gray; padding: 5px; margin: 10px 0;"> <pre>RESERVATIONPOLICY HIGHEST RESERVATIONDEPTH 3</pre> </div> <p>Only jobs 20, 12, and 10 get priority reservations. Later, if a job with priority higher than 20 is submitted into the queue, it will also get a priority reservation along with the jobs listed previously. If four jobs higher than 20 were to be submitted into the queue, only three would get priority reservations, in accordance with the condition set in the <b>RESERVATIONDEPTH</b> policy.</p> <p>With <b>HIGHEST</b>, Moab may appear to exceed the <b>RESERVATIONDEPTH</b> if it has already scheduled the maximum number of priority reservations and then users submit jobs with higher priority than those already given a priority reservation. Moab keeps all of the previously-created priority reservations and creates new ones for jobs with higher priority (again up to the quantity specified with <b>RESERVATIONDEPTH</b>). This means that, if your <b>RESERVATIONDEPTH</b> is set to 3, Moab can potentially schedule up to 3 new priority reservations each scheduling iteration, as long as new higher-priority jobs are continually submitted. This behavior ensures that the highest-priority jobs receive attention while the former highest-priority jobs do not lose their priority reservation.</p>
<b>CURRENTHIGHEST</b>	<p>Only the current top &lt;<b>RESERVATIONDEPTH</b>&gt; priority jobs receive reservations. Under this policy, all job reservations are destroyed each iteration when the queue is re-prioritized. The top jobs in the queue are then given new reservations.</p>
<b>NEVER</b>	<p>No priority reservations are made.</p>

## Priority Reservation Depth

By default, only the highest priority job receives a priority reservation. However, this behavior is configurable via the [RESERVATIONDEPTH](#) policy. Moab's default behavior of only reserving the highest priority job allows backfill to be used in a form known as liberal backfill. Liberal backfill tends to maximize system utilization and minimize overall average job turnaround time. However, it does lead to the potential of some lower priority jobs being indirectly delayed and may lead to greater variance in job turnaround time. The **RESERVATIONDEPTH** parameter can be set to a very large value, essentially enabling what is called conservative backfill where every job that cannot run is given a reservation. Most sites prefer the liberal backfill approach associated with the default **RESERVATIONDEPTH** of 1 or else select a slightly higher value. It is important to note that to prevent starvation in conjunction with

reservations, monotonically increasing priority factors such as queue time or job XFactor should be enabled. See the [Prioritization Overview](#) for more information on priority factors.

Another important consequence of backfill and reservation depth is how they affect job priority. In Moab, all jobs are prioritized. Backfill allows jobs to be run out of order and thus, to some extent, job priority to be ignored. This effect, known as priority dilution, can cause many site policies implemented via Moab prioritization policies to be ineffective. Setting the **RESERVATIONDEPTH** parameter to a higher value gives job priority more teeth at the cost of slightly lower system utilization. This lower utilization results from the constraints of these additional reservations, decreasing the scheduler's freedom and its ability to find additional optimizing schedules. Anecdotal evidence indicates that these utilization losses are fairly minor, rarely exceeding 8%.

It is difficult a priori to know the right setting for the **RESERVATIONDEPTH** parameter. Surveys indicate that the vast majority of sites use the default value of 1. Sites that do modify this value typically set it somewhere in the range of 2 to 10. The following guidelines may be useful in determining if and how to adjust this parameter:

### Reasons to Increase RESERVATIONDEPTH

- The estimated job start time information provided by the [showstart](#) command is heavily used and the accuracy needs to be increased.
- Priority dilution prevents certain key mission objectives from being fulfilled.
- Users are more interested in knowing when their job will run than in having it run sooner.

### Reasons to Decrease RESERVATIONDEPTH

- Scheduling efficiency and job throughput need to be increased.

### Assigning Per-QoS Reservation Creation Rules

QoS based reservation depths can be enabled via the [RESERVATIONQOSLIST](#) parameter. This parameter allows varying reservation depths to be associated with different sets of job QoSs. For example, the following configuration creates two reservation depth groupings:

```
RESERVATIONDEPTH[0]      8
RESERVATIONQOSLIST[0]   highprio, interactive, debug
RESERVATIONDEPTH[1]      2
RESERVATIONQOSLIST[1]   batch
```

*This example causes that the top 8 jobs belonging to the aggregate group of **highprio**, **interactive**, and **debug** QoS jobs will receive priority reservations. Additionally, the top two **batch** QoS jobs will also receive priority reservations. Use of this feature allows sites to maintain high throughput for important jobs by guaranteeing that a significant proportion of these jobs progress toward starting through use of the priority reservation.*

By default, the following parameters are set inside Moab:

```
RESERVATIONDEPTH[DEFAULT] 1
RESERVATIONQOSLIST[DEFAULT] ALL
```

*This allows one job with the highest priority to get a reservation. These values can be overwritten by modifying the **DEFAULT** policy.*

## Managing Resource Failures

Moab allows organizations to control how to best respond to a number of real-world issues. Occasionally when a reservation becomes active and a job attempts to start, various resource manager race conditions or corrupt state situations will prevent the job from starting. By default, Moab assumes the resource manager is corrupt, releases the reservation, and attempts to re-create the reservation after a short timeout. However, in the interval between the reservation release and the re-creation timeout, other priority reservations may allocate the newly available resources, reserving them before the original reservation gets an opportunity to reallocate them. Thus, when the original job reservation is re-established, its original resource may be unavailable and the resulting new reservation may be delayed several hours from the earlier start time. The parameter [RESERVATIONRETRYTIME](#) allows a site that is experiencing frequent resource manager race conditions and/or corruption situations to tell Moab to hold on to the reserved resource for a period of time in an attempt to allow the resource manager to correct its state.

## Resource Allocation Policy

By default, when a standing or administrative reservation is created, Moab allocates nodes in accordance with the specified taskcount, node expression, node constraints, and the [MINRESOURCE](#) node allocation policy.

### Related topics

- [Reservation Overview](#)
- [Backfill](#)

## Configuring and Managing Reservations

- [Reservation Attributes](#)
  - [Start/End Time](#)
  - [Access Control List \(ACL\)](#)
  - [Selecting Resources](#)
  - [Flags](#)
- [Configuring and Managing Standing Reservations](#)
  - [Standing Reservation Attributes](#)
  - [Standing Reservation Overview](#)
  - [Specifying Reservation Resources](#)
  - [Enforcing Policies Via Multiple Reservations](#)
  - [Affinity](#)
  - [ACL Modifiers](#)
  - [Reservation Ownership](#)

- [Partitions](#)
- [Resource Allocation Behavior](#)
- [Rolling Reservations](#)
- [Modifying Resources with Standing Reservations](#)
- [Managing Administrative Reservations](#)

## Reservation Attributes

All reservations possess a time frame of activity, an access control list (ACL), and a list of resources to be reserved. Additionally, reservations may also possess a number of extension attributes including epilog/prolog specification, reservation ownership and accountability attributes, and special flags that modify the reservation's behavior.

### Start/End Time

All reservations possess a start and an end time that define the reservation's active time. During this active time, the resources within the reservation may only be used as specified by the reservation access control list (ACL). This active time may be specified as either a start/end pair or a start/duration pair. Reservations exist and are visible from the time they are created until the active time ends at which point they are automatically removed.

### Access Control List (ACL)

For a reservation to be useful, it must be able to limit who or what can access the resources it has reserved.

**i** By default a reservation may allocate resources that possess credentials that meet the submitter's ACL. In other words, a user's reservation won't necessarily allocate only free and idle nodes. If a reservation exists that coincides with the submitter's ACL, the nodes under that reservation are also considered for allocation. This is referred to as ACL overlap. To make new reservations allocate *only* free and idle nodes, you must use the [NOACLOVERLAP](#) flag.

This is handled by way of an ACL. With reservations, ACLs can be based on credentials, resources requested, or performance metrics. In particular, with a standing reservation, the attributes [USERLIST](#), [GROUPLIST](#), [ACCOUNTLIST](#), [CLASSLIST](#), [QOSLIST](#), [JOBATTRLIST](#), [PROCLIMIT](#), [MAXTIME](#), or [TIMELIMIT](#) may be specified. (See [Affinity](#) and [Modifiers](#).)

**i** Reservation access can be adjusted based on a job's requested node features by mapping node feature requests to job attributes as in the following example:

```
NODECFG[DEFAULT]  FEATURES=ia64
NODETOJOBATTRMAP  ia64, ia32
SRCFG[pgs]        JOBATTRLIST=ia32
```

```
> mrsvctl -c -a jattr=gpfs\! -h "r:13-500"
```

## Selecting Resources

When specifying which resources to reserve, the administrator has a number of options. These options allow control over how many resources are reserved and where they are reserved. The following reservation attributes allow the administrator to define resources.

### Task Description

Moab uses the task concept extensively for its job and reservation management. A task is simply an atomic collection of resources, such as processors, memory, or local disk, which must be found on the same node. For example, if a task requires 4 processors and 2 GB of memory, the scheduler must find all processors AND memory on the same node; it cannot allocate 3 processors and 1 GB on one node and 1 processor and 1 GB of memory on another node to satisfy this task. Tasks constrain how the scheduler must collect resources for use in a standing reservation; however, they do not constrain the way in which the scheduler makes these cumulative resources available to jobs. A job can use the resources covered by an accessible reservation in whatever way it needs. If reservation X allocates 6 tasks with 2 processors and 512 MB of memory each, it could support job Y which requires 10 tasks of 1 processor and 128 MB of memory or job Z which requires 2 tasks of 4 processors and 1 GB of memory each. The task constraints used to acquire a reservation's resources are transparent to a job requesting use of these resources.

Example 3-103:

```
SRCFG[test] RESOURCES=PROCS:2, MEM:1024
```

### Taskcount

Using the task description, the taskcount attribute defines how many tasks must be allocated to satisfy the reservation request. To create a reservation, a taskcount and/or a hostlist must be specified.

Example 3-104:

```
SRCFG[test] TASKCOUNT=256
```

### Hostlist

A hostlist constrains the set of resources available to a reservation. If no taskcount is specified, the reservation attempts to reserve one task on each of the listed resources. If a taskcount is specified that requests fewer resources than listed in the hostlist, the scheduler reserves only the number of tasks from the hostlist specified by the taskcount attribute. If a taskcount is specified that requests more resources than listed in the hostlist, the scheduler reserves the hostlist nodes first and then seeks additional resources outside of this list.

Example 3-105:

```
SRCFG[test] HOSTLIST=node01,node1[3-5]
```

### Node Features

Node features can be specified to constrain which resources are considered.

Example 3-106:

```
SRCFG[test] NODEFEATURES=fastos
```

### Partition

A partition may be specified to constrain which resources are considered.

Example 3-107:

```
SRCFG[test] PARTITION=core3
```

## Flags

Reservation flags allow specification of special reservation attributes or behaviors. Supported flags are listed in the following table:

Flag Name	Description
<b>ACLOVERLAP</b>	<b>Deprecated (this is now a default flag).</b> In addition to free or idle nodes, a reservation may also reserve resources that possess credentials that meet the reservation's ACL. To change this behavior, set the <a href="#">NOACLOVERLAP on page 465</a> flag.
<b>ADVRESJOBDESTROY</b>	All jobs that have an <a href="#">ADVRES</a> matching this reservation are canceled when the reservation is destroyed.
<b>ALLOWJOBOVERLAP</b>	A job is allowed to start in a reservation that may end before the job completes. When the reservation ends before the job completes, the job will not be canceled but will continue to run.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job <a href="#">ADVRES</a> flag. (See <a href="#">Job to Reservation Binding</a> .)

Flag Name	Description
<p><b>DEDICATEDRESOURCE</b> (aka <b>EXCLUSIVE</b>)</p>	<p>Reservation placed only on resources that are not reserved by any other reservation including job, system, and user reservation. There are two exception to this:</p> <ol style="list-style-type: none"> <li>1. Reserved resources could be allocated when <b>DEDICATEDRESOURCE</b> is combined with <a href="#">IGNJOBRSV*</a></li> <li>2. Reserved resources could be allocated when a reservation matches the submitter's ACL. In this case, to make <b>DEDICATEDRESOURCE</b> <i>truly</i> exclusive, use the <a href="#">NOACLOVERLAP</a> flag.</li> </ol> <div data-bbox="646 590 1409 779" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> The order that SRCFG reservations are listed in the configuration is important when using <b>DEDICATEDRESOURCE</b>, because reservations made afterwards can steal resources later. During configuration, list <b>DEDICATEDRESOURCE</b> reservations last to guarantee exclusiveness.</p> </div>
<p><b>EVACVMS</b></p>	<p>Reservation will automatically evacuate virtual machines from the reservation nodelist.</p> <div data-bbox="646 898 1409 1024" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> The same action can be accomplished by using reservation profiles. For more information, see <a href="#">Optimizing Maintenance Reservations</a> on page 455.</p> </div>
<p><b>IGNIDLEJOBS*</b></p>	<p>Reservation can be placed on top of idle job reservations.</p> <div data-bbox="646 1108 1409 1203" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> This flag is meant to be used in conjunction with <b>DEDICATEDRESOURCE</b>.</p> </div>
<p><b>IGNJOBRSV*</b></p>	<p>Ignores existing job reservations, allowing the reservation to be forced onto available resources even if it conflicts with existing job reservations. User and system reservation conflicts are still valid. It functions the same as <a href="#">IGNIDLEJOBS</a> plus allows a reservation to be placed on top of an existing running job's reservation.</p> <div data-bbox="646 1413 1409 1507" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> This flag is meant to be used in conjunction with <b>DEDICATEDRESOURCE</b>.</p> </div>
<p><b>IGNRSV*</b></p>	<p>Request ignores existing resource reservations allowing the reservation to be forced onto available resources even if this conflicts with other reservations. It functions the same as <a href="#">IGNJOBRSV</a> plus allows the reservation to be placed on top of the system reservations.</p> <div data-bbox="646 1686 1409 1780" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> This flag is meant to be used in conjunction with <b>DEDICATEDRESOURCE</b>.</p> </div>

Flag Name	Description
<b>IGNSTATE*</b>	Reservation ignores node state when assigning nodes. It functions the same as <a href="#">IGNRSV</a> plus allows the reservation to be placed on nodes that are not currently available. Also ignores resource availability on nodes.
<b>NOACLOVERLAP</b>	<p>All resources must be free or idle, with no existing reservations. Moab will not allocate in-use resources even if they match the reservation's ACL.</p> <pre>mrsvctl -c -t 12 -E -F noaclovelap -a user==john</pre> <p><i>Moab looks for resources that are exclusive (free). Without the flag, Moab would look for resources that are exclusive or that are already running john's jobs.</i></p> <p><b>i</b> This flag is meant to be used in conjunction with <a href="#">DEDICATEDRESOURCE</a>.</p>
<b>NOVMIGRATION</b>	If set on a reservation, this prevents VMs from being migrated away from the reservation. If there are multiple reservations on the hypervisor and at least one reservation does not have the <a href="#">NOVMIGRATION</a> flag, then VMs will be migrated.
<b>OWNERPREEMPT</b>	Jobs by the reservation <a href="#">owner</a> are allowed to preempt non-owner jobs using reservation resources.
<b>OWNERPREEMPTIGNOREMINTIME</b>	<p>Allows the <a href="#">OWNERPREEMPT</a> flag to "trump" the <a href="#">PREEMPTMINTIME</a> setting for jobs already running on a reservation when the owner of the reservation submits a job. For example: without the <a href="#">OWNERPREEMPTIGNOREMINTIME</a> flag set, a job submitted by the owner of a reservation will not preempt non-owner jobs already running on the reservation until the <a href="#">PREEMPTMINTIME</a> setting (if set) for those jobs is passed.</p> <p>With the <a href="#">OWNERPREEMPTIGNOREMINTIME</a> flag set, a job submitted by the owner of a reservation immediately preempts non-owner jobs already running on the reservation, regardless of whether <a href="#">PREEMPTMINTIME</a> is set for the non-owner jobs.</p>
<b>REQFULL</b>	Reservation is only created when all resources can be allocated.
<b>SINGLEUSE</b>	Reservation is automatically removed after completion of the first job to use the reserved resources.

Flag Name	Description
SPACEFLEX	<b>Deprecated (this is now a default flag).</b> Reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.

**i** \* *IGNIDLEJOBS*, *IGNJOBRSV*, *IGNRSV*, and *IGNSTATE* flags are built on one another and form a hierarchy. *IGNJOBRSV* performs the function of *IGNIDLEJOBS* plus its own functions. *IGNRSV* performs the function of *IGNJOBRSV* and *IGNIDLEJOBS* plus its own functions. *IGNSTATE* performs the function of *IGNRSV*, *IGNJOBRSV*, and *IGNIDLEJOBS* plus its own functions. While you can use combinations of these flags, it is not necessary. If you set one flag, you do not need to set other flags that fall beneath it in the hierarchy.

Most flags can be associated with a reservation via the [mrsvctl -c -F](#) command or the [SRCFG](#) parameter.

## Configuring Standing Reservations

Standing reservations allow resources to be dedicated for particular uses. This dedication can be configured to be permanent or periodic, recurring at a regular time of day and/or time of week. There is extensive applicability of standing reservations for everything from daily dedicated job runs to improved use of resources on weekends. By default, standing reservations can overlap other reservations. Unless you set an ignore-type flag (*ACLOVERLAP*, *DEDICATEDRESOURCE*, *IGNIDLEJOBS*, or *IGNJOBRSV*), they are automatically given the *IGNRSV* flag. All standing reservation attributes are specified via the [SRCFG](#) parameter using the attributes listed in the table below.

### Standing Reservation Attributes

ACCESS	
<b>Format</b>	<i>DEDICATED</i> or <i>SHARED</i>
<b>Default</b>	---
<b>Description</b>	If set to <i>SHARED</i> , allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.
<b>Example</b>	<pre>SRCFG[test] ACCESS=SHARED</pre> <p><i>Standing reservation test may access resources allocated to existing standing and administrative reservations.</i></p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> The order that SRCFG reservations are listed in the configuration are important when using <i>DEDICATED</i>, because reservations made afterwards can steal resources later. During configuration, list <i>DEDICATED</i> reservations last to guarantee exclusiveness.</p> </div>

ACCOUNTLIST	
<b>Format</b>	List of valid, comma delimited account names (see <a href="#">ACL Modifiers</a> ).
<b>Default</b>	---
<b>Description</b>	Specifies that jobs with the associated accounts may use the resources contained within this reservation.
<b>Example</b>	<pre>SRCFG[test] ACCOUNTLIST=ops,staff</pre> <p><i>Jobs using the account <b>ops</b> or <b>staff</b> are granted access to the resources in standing reservation <b>test</b>.</i></p>

CHARGE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	Overrides the default charging behavior. If set to <i>True</i> , indicates that this reservation should be charged, even if no <b>ChargeAccount</b> or <b>ChargeUser</b> are specified (this assumes your Accounting Manager is set up to permit this). If set to <i>False</i> , indicates that this reservation should not be charged. It is not necessary to specify <b>CHARGE=True</b> if <b>CHARGEACCOUNT</b> or <b>CHARGEUSER</b> is specified.
<b>Example</b>	<pre>SRCFG[sr_gold1] CHARGE=False</pre> <p><i>Prevent charges to this reservation (might be used when <b>AMCFG[] ALWAYSCHARGERESERVATIONS=True</b>).</i></p>

CHARGEACCOUNT	
<b>Format</b>	Any valid account name.
<b>Default</b>	---
<b>Description</b>	Specifies that idle cycles for this reservation should be charged against the specified account (via the Accounting Manager).
	<div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; display: inline-block;">  <b>CHARGEACCOUNT</b> must be used in conjunction with <a href="#">CHARGEUSER</a>.         </div>

## CHARGEACCOUNT

<b>Example</b>	<pre>SRCFG[sr_gold1] CHARGEACCOUNT=math SRCFG[sr_gold1] CHARGEUSER=john</pre> <p><i>Moab charges all idle cycles within reservations supporting standing reservation <code>sr_gold1</code> to account <code>math</code>.</i></p>
----------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## CHARGEUSER

<b>Format</b>	Any valid username.
<b>Default</b>	---
<b>Description</b>	<p>Specifies that idle cycles for this reservation should be charged against the specified user (via the Accounting Manager).</p> <div style="border: 1px solid #0070c0; border-radius: 5px; padding: 5px; background-color: #e6f2ff;">  <b>CHARGEUSER</b> must be used in conjunction with <a href="#">CHARGEACCOUNT</a>.         </div>
<b>Example</b>	<pre>SRCFG[sr_gold1] CHARGEACCOUNT=math SRCFG[sr_gold1] CHARGEUSER=john</pre> <p><i>Moab charges all idle cycles within reservations supporting standing reservation <code>sr_gold1</code> to user <code>john</code>.</i></p>

## CLASSLIST

<b>Format</b>	List of valid, comma delimited classes/queues (see <a href="#">ACL Modifiers</a> ).
<b>Default</b>	---
<b>Description</b>	Specifies that jobs with the associated classes/queues may use the resources contained within this reservation.
<b>Example</b>	<pre>SRCFG[test] CLASSLIST=!interactive</pre> <p><i>Jobs not using the class <code>interactive</code> are granted access to the resources in standing reservation <code>test</code>.</i></p>

CLUSTERLIST	
<b>Format</b>	List of valid, comma-delimited peer clusters (see <a href="#">Moab Workload Manager for Grids</a> ).
<b>Default</b>	---
<b>Description</b>	Specifies that jobs originating within the listed clusters may use the resources contained within this reservation.
<b>Example</b>	<pre>SRCFG[test] CLUSTERLIST=orion2,orion7</pre> <p><i>Moab grants jobs from the listed peer clusters access to the reserved resources.</i></p>

COMMENT	
<b>Format</b>	<STRING>  If the string contains whitespace, it should be enclosed in single (') or double quotes (").
<b>Default</b>	---
<b>Description</b>	Specifies a descriptive message associated with the standing reservation and all child reservations.
<b>Example</b>	<pre>SRCFG[test] COMMENT='rsv for network testing'</pre> <p><i>Moab annotates the standing reservation <b>test</b> and all child reservations with the specified message. These messages show up within Moab client commands, Moab web tools, and graphical administrator tools.</i></p>

DAYS	
<b>Format</b>	One or more of the following (comma-delimited): <ul style="list-style-type: none"> <li>• <i>Mon</i></li> <li>• <i>Tue</i></li> <li>• <i>Wed</i></li> <li>• <i>Thu</i></li> <li>• <i>Fri</i></li> <li>• <i>Sat</i></li> <li>• <i>Sun</i></li> <li>• <i>[ALL]</i></li> </ul>

DAYS	
<b>Default</b>	<i>[ALL]</i>
<b>Description</b>	Specifies which days of the week the standing reservation is active.
<b>Example</b>	<pre>SRCFG[test] DAYS=Mon,Tue,Wed,Thu,Fri</pre> <p><i>Standing reservation <b>test</b> is active Monday through Friday.</i></p>

DEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	2
<b>Description</b>	<p>Specifies the depth of standing reservations to be created (one per <a href="#">period</a>).</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> To satisfy the <b>DEPTH</b>, Moab creates new reservations at the beginning of the specified <a href="#">PERIOD</a> on page 475. If your reservation ends at the same time that a new <b>PERIOD</b> begins, the number of reservations may not match the requested <b>DEPTH</b>. To prevent or resolve this issue, set the <a href="#">ENDTIME</a> on page 471 a couple minutes before the beginning of the next <b>PERIOD</b>. For example, set the <b>ENDTIME</b> to <i>23:58</i> instead of <i>00:00</i>.</p> </div>
<b>Example</b>	<pre>SRCFG[test] PERIOD=DAY DEPTH=6</pre> <p><i>Specifies that six reservations will be created for standing reservation <b>test</b>.</i></p>

DISABLE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies that the standing reservation should no longer spawn child reservations.
<b>Example</b>	<pre>SRCFG[test] PERIOD=DAY DEPTH=7 DISABLE=TRUE</pre> <p><i>Specifies that reservations are created for standing reservation <b>test</b> for today and the next six days.</i></p>

ENDTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	24:00:00
<b>Description</b>	Specifies the time of day the standing reservation period ends (end of day or end of week depending on <a href="#">PERIOD</a> ).
<b>Example</b>	<pre>SRCFG[test] STARTTIME=8:00:00 SRCFG[test] ENDTIME=17:00:00 SRCFG[test] PERIOD=DAY</pre> <p><i>Standing reservation <b>test</b> is active from 8:00 AM until 5:00 PM.</i></p>

FLAGS	
<b>Format</b>	Comma-delimited list of zero or more flags listed in the <a href="#">reservation flags overview</a> .
<b>Default</b>	---
<b>Description</b>	Specifies special reservation attributes. See <a href="#">Managing Reservations - Flags</a> for details.
<b>Example</b>	<pre>SRCFG[test] FLAGS=BYNAME, DEDICATEDRESOURCE</pre> <p><i>Jobs may only access the resources within this reservation if they explicitly request the reservation by name. Further, the reservation is created to not overlap with other reservations.</i></p>

GROUPLIST	
<b>Format</b>	One or more comma-delimited group names.
<b>Default</b>	[ALL]
<b>Description</b>	Specifies the groups allowed access to this standing reservation (see <a href="#">ACL Modifiers</a> ).
<b>Example</b>	<pre>SRCFG[test] GROUPLIST=staff, ops, special SRCFG[test] CLASSLIST=interactive</pre> <p><i>Moab allows jobs with the listed group IDs or which request the job class <b>interactive</b> to use the resources covered by the standing reservation.</i></p>

HOSTLIST	
<b>Format</b>	One or more comma delimited host names or <a href="#">host expressions</a> or the string "class:<classname>".
<b>Default</b>	---
<b>Description</b>	<p>Specifies the set of hosts that the scheduler can search for resources to satisfy the reservation. If specified using the "class:X" format, Moab only selects hosts that support the specified class. If <b>TASKCOUNT</b> is also specified, only <b>TASKCOUNT</b> tasks are reserved. Otherwise, all matching hosts are reserved.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p> The <b>HOSTLIST</b> attribute is treated as host regular expression so <code>foo10</code> will map to <code>foo10</code>, <code>foo101</code>, <code>foo1006</code>, and so forth. To request an exact host match, the expression can be bounded by the caret and dollar symbol expression markers as in <code>^foo10\$</code>.</p> </div>
<b>Example</b>	<pre>SRCFG[test] HOSTLIST=node001,node002,node003 SRCFG[test] RESOURCES=PROCS:2;MEM:512 SRCFG[test] TASKCOUNT=2</pre> <p><i>Moab reserves a total of two tasks with 2 processors and 512 MB each, using resources located on <b>node001</b>, <b>node002</b>, and/or <b>node003</b>.</i></p> <pre>SRCFG[test] HOSTLIST=node01,node1[3-5]</pre> <p><i>The reservation will consume all nodes that have "node01" somewhere in their names and all nodes that have both "node1" and either a "3," "4," or "5" in their names.</i></p> <pre>SRCFG[test] HOSTLIST=r:node[1-6]</pre> <p><i>The reservation will consume all nodes with names that begin with "node" and end with any number 1 through 6. In other words, it will reserve <b>node1</b>, <b>node2</b>, <b>node3</b>, <b>node4</b>, <b>node5</b>, and <b>node6</b>.</i></p>

JOBATTRLIST	
<b>Format</b>	Comma-delimited list of one or more of the following job attributes: <ul style="list-style-type: none"> <li>• <b>PREEMPTEE</b></li> <li>• <b>INTERACTIVE</b></li> <li>• any generic attribute configured through <b>NODECFG</b>.</li> </ul>
<b>Default</b>	---

JOBATTRLIST	
<b>Description</b>	<p>Specifies job attributes that grant a job access to the reservation.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-bottom: 5px;"> <p><b>i</b> Values can be specified with a "!=" assignment to only allow jobs NOT requesting a certain feature inside the reservation.</p> </div> <div style="border: 1px solid #0070C0; padding: 5px;"> <p><b>i</b> To enable/disable reservation access based on requested node features, use the parameter <a href="#">NODETOJOBATTRMAP</a>.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #000; padding: 5px; margin-bottom: 5px;"> <pre>SRCFG[test] JOBATTRLIST=PREEMPTEE</pre> </div> <div style="border: 1px dashed #000; padding: 5px;"> <p><i>Preemptible jobs can access the resources reserved within this reservation.</i></p> </div>

MAXJOB	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Specifies the maximum number of jobs that can run in the reservation.
<b>Example</b>	<div style="border: 1px dashed #000; padding: 5px; margin-bottom: 5px;"> <pre>SRCFG[test] MAXJOB=1</pre> </div> <div style="border: 1px dashed #000; padding: 5px;"> <p><i>Only one job will be allowed to run in this reservation.</i></p> </div>

MAXTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS[+]
<b>Default</b>	---
<b>Description</b>	Specifies the maximum time for jobs allowable. Can be used with Affinity to attract jobs with same <b>MAXTIME</b> .
<b>Example</b>	<div style="border: 1px dashed #000; padding: 5px; margin-bottom: 5px;"> <pre>SRCFG[test] MAXTIME=1:00:00+</pre> </div> <div style="border: 1px dashed #000; padding: 5px;"> <p><i>Jobs with a time of 1:00:00 are attracted to this reservation.</i></p> </div>

NODEFEATURES	
<b>Format</b>	Comma-delimited list of node features.
<b>Default</b>	---
<b>Description</b>	Specifies the required node features for nodes that are part of the standing reservation.
<b>Example</b>	<pre>SRCFG[test] NODEFEATURES=wide, fddi</pre> <p><i>All nodes allocated to the standing reservation must have both the <b>wide</b> and <b>fddi</b> node attributes.</i></p>

OWNER	
<b>Format</b>	<p>&lt;CREDTYPE&gt;:&lt;CREDID&gt;</p> <p>Where &lt;CREDTYPE&gt; is one of <b>USER</b>, <b>GROUP</b>, <b>ACCT</b>, <b>QoS</b>, <b>CLASS</b> or <b>CLUSTER</b> and &lt;CREDTYPE&gt; is a valid credential id of that type.</p>
<b>Default</b>	---
<b>Description</b>	<p>Specifies the owner of the reservation. Setting ownership for a reservation grants the user management privileges, including the power to release it.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-bottom: 5px;"> <p><b>i</b> Setting a <b>USER</b> as the <b>OWNER</b> of a reservation gives that user privileges to query and release the reservation.</p> </div> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px;"> <p><b>i</b> For sandbox reservations, sandboxes are applied to a specific peer only if <b>OWNER</b> is set to <b>CLUSTER:&lt;PEERNAME&gt;</b>.</p> </div>
<b>Example</b>	<pre>SRCFG[test] OWNER=ACCT:jupiter</pre> <p><i>User <b>jupiter</b> owns the reservation and may be granted special privileges associated with that ownership.</i></p>

PARTITION	
<b>Format</b>	Valid partition name.
<b>Default</b>	<b>[ALL]</b>

PARTITION	
<b>Description</b>	Specifies the partition in which to create the standing reservation.
<b>Example</b>	<pre>SRCFG[test] PARTITION=OLD</pre> <p><i>The standing reservation will only select resources from partition <b>OLD</b>.</i></p>

PERIOD	
<b>Format</b>	One of <i>DAY</i> , <i>WEEK</i> , or <i>INFINITY</i> .
<b>Default</b>	<i>DAY</i>
<b>Description</b>	Specifies the period of the standing reservation.
<b>Example</b>	<pre>SRCFG[test] PERIOD=WEEK</pre> <p>Each standing reservation covers a one week period.</p>

PROCLIMIT	
<b>Format</b>	<p>&lt;QUALIFIER&gt;&lt;INTEGER&gt;          &lt;QUALIFIER&gt; may be one of the following &lt;, &lt;=, ==, &gt;=, &gt;</p>
<b>Default</b>	---
<b>Description</b>	Specifies the processor limit for jobs requesting access to this standing reservation.
<b>Example</b>	<pre>SRCFG[test] PROCLIMIT&lt;=4</pre> <p><i>Jobs requesting 4 or fewer processors are allowed to run.</i></p>

PSLIMIT	
<b>Format</b>	<p>&lt;QUALIFIER&gt;&lt;INTEGER&gt;          &lt;QUALIFIER&gt; may be one of the following &lt;, &lt;=, ==, &gt;=, &gt;</p>

PSLIMIT	
<b>Default</b>	---
<b>Description</b>	Specifies the processor-second limit for jobs requesting access to this standing reservation.
<b>Example</b>	<pre>SRCFG[test] PSLIMIT&lt;=40000</pre> <p><i>Jobs requesting 40000 or fewer processor-seconds are allowed to run.</i></p>

QOSLIST	
<b>Format</b>	Zero or more valid, comma-delimited QoS names.
<b>Default</b>	---
<b>Description</b>	Specifies that jobs with the listed QoS names can access the reserved resources.
<b>Example</b>	<pre>SRCFG[test] QOSLIST=hi, low, special</pre> <p><i>Moab allows jobs using the listed QoS's access to the reserved resources.</i></p>

REQUIREDTPN	
<b>Format</b>	<p>&lt;QUALIFIER&gt;&lt;INTEGER&gt;</p> <p>&lt;QUALIFIER&gt; may be one of the following &lt;, &lt;=, ==, &gt;=, &gt;</p>
<b>Default</b>	---
<b>Description</b>	Restricts access to reservations based on the job's TPN (tasks per node).
<b>Example</b>	<pre>SRCFG[test] REQUIREDTPN==4</pre> <p><i>Jobs with tpn=4 or ppn=4 would be allowed within the reservation, but any other TPN value would not. (For more information, see <a href="#">TPN (Exact Tasks Per Node)</a> on page 479.)</i></p>

RESOURCES	
<b>Format</b>	Semicolon delimited <ATTR>:<VALUE> pairs where <ATTR> may be one of <i>PROCS</i> , <i>MEM</i> , <i>SWAP</i> , or <i>DISK</i> .
<b>Default</b>	<i>PROCS</i> :-1 (All processors available on node)
<b>Description</b>	<p>Specifies what resources constitute a single standing reservation task. (Each task must be able to obtain all of its resources as an atomic unit on a single node.) Supported resources currently include the following:</p> <ul style="list-style-type: none"> <li>• <i>PROCS</i> (number of processors)</li> <li>• <i>MEM</i> (real memory in MB)</li> <li>• <i>DISK</i> (local disk in MB)</li> <li>• <i>SWAP</i> (virtual memory in MB)</li> </ul>
<b>Example</b>	<pre>SRCFG[test] RESOURCES=PROCS:1;MEM:512</pre> <p><i>Each standing reservation task reserves one processor and 512 MB of real memory.</i></p>

ROLLBACKOFFSET	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	<p>Specifies the minimum time in the future at which the reservation may start. This offset is rolling meaning the start time of the reservation will continuously roll back into the future to maintain this offset. Rollback offsets are a good way of providing guaranteed resource access to users under the conditions that they must commit their resources in the future or lose dedicated access. See <a href="#">QoS</a> for more info about quality of service and service level agreements; also see <a href="#">Rollback Reservation Overview</a>.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF;"> <p> Neither credlock nor advres is compatible on the jobs submitted for this reservation.</p> </div>
<b>Example</b>	<pre>SRCFG[ajax] ROLLBACKOFFSET=24:00:00 TASKCOUNT=32 SRCFG[ajax] PERIOD=INFINITY ACCOUNTLIST=ajax</pre> <p><i>The standing reservation guarantees access to up to 32 processors within 24 hours to jobs from the ajax account.</i></p> <p>Adding an asterisk to the <b>ROLLBACKOFFSET</b> value pins rollback reservation start times when an idle reservation is created in the rollback reservation. For example:</p> <pre>SRCFG[staff] ROLLBACKOFFSET=18:00:00* PERIOD=INFINITY</pre>

RSVACCESSLIST	
<b>Format</b>	<RESERVATION>[,...]
<b>Default</b>	---
<b>Description</b>	A list of reservations to which the specified reservation has access.
<b>Example</b>	<pre>SRCFG[test] RSVACCESSLIST=rsv1,rsv2,rsv3</pre>

RSVGROUP	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	See section <a href="#">Reservation Group</a> for a detailed description.
<b>Example</b>	<pre>SRCFG[test] RSVGROUP=rsvgrp1 SRCFG[ajax] RSVGROUP=rsvgrp1</pre>

STARTTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	00:00:00:00 (midnight)
<b>Description</b>	<p>Specifies the time of day/week the standing reservation becomes active. Whether this indicates a time of day or time of week depends on the setting of the <a href="#">PERIOD</a> attribute.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #D9E1F2;"> <p><b>i</b> If specified within a <a href="#">reservation profile</a>, a value of 0 indicates the reservation should start at the earliest opportunity.</p> </div>
<b>Example</b>	<pre>SRCFG[test] STARTTIME=08:00:00 SRCFG[test] ENDTIME=17:00:00 SRCFG[test] PERIOD=DAY</pre> <p><i>The standing reservation will be active from 8:00 a.m. until 5:00 p.m. each day.</i></p>

TASKCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (unlimited tasks)
<b>Description</b>	Specifies how many tasks should be reserved for the reservation.
<b>Example</b>	<pre>SRCFG[test] RESOURCES=PROCS:1;MEM:256 SRCFG[test] TASKCOUNT=16</pre> <p><i>Standing reservation <b>test</b> reserves 16 tasks worth of resources; in this case, 16 processors and 4 GB of real memory.</i></p>

TIMELIMIT	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	-1 (no time based access)
<b>Description</b>	Specifies the maximum allowed overlap between the standing reservation and a job requesting resource access.
<b>Example</b>	<pre>SRCFG[test] TIMELIMIT=1:00:00</pre> <p><i>Moab allows jobs to access up to one hour of resources in the standing reservation.</i></p>

TPN (Exact Tasks Per Node)	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (no TPN constraint)
<b>Description</b>	Specifies the exact number of tasks per node that must be available on eligible nodes.
<b>Example</b>	<pre>SRCFG[2] TPN=4 SRCFG[2] RESOURCES=PROCS:2;MEM:256</pre> <p><i>Moab must locate four tasks on each node that is to be part of the reservation. That is, each node included in standing reservation 2 must have 8 processors and 1 GB of memory available.</i></p>

TRIGGER	
<b>Format</b>	See <a href="#">Creating a trigger</a> on page 748 for syntax.
<b>Default</b>	N/A
<b>Description</b>	Specifies event triggers to be launched by the scheduler under the scheduler's ID. These triggers can be used to conditionally cancel reservations, <a href="#">modify resources</a> , or launch various actions at specified event offsets. See <a href="#">About object triggers</a> on page 745 for more detail.
<b>Example</b>	<pre>SRCFG[fast] TRIGGER=EType=start,Offset=5:00:00,AType=exec,Action="/usr/local/domail.pl"</pre> <p><i>Moab launches the domail.pl script 5 hours after any fast reservation starts.</i></p>

USERLIST	
<b>Format</b>	Comma-delimited list of users.
<b>Default</b>	---
<b>Description</b>	Specifies which users have access to the resources reserved by this reservation (see <a href="#">ACL Modifiers</a> ).
<b>Example</b>	<pre>SRCFG[test] USERLIST=bob,joe,mary</pre> <p><i>Users bob, joe and mary can all access the resources reserved within this reservation.</i></p>

### Standing Reservation Overview

A standing reservation is similar to a normal administrative reservation in that it also places an access control list on a specified set of resources. Resources are specified on a per-task basis and currently include processors, local disk, real memory, and swap. The access control list supported for standing reservations includes users, groups, accounts, job classes, and QoS levels. Standing reservations can be configured to be permanent or periodic on a daily or weekly basis and can accept a daily or weekly start and end time. Regardless of whether permanent or recurring on a daily or weekly basis, standing reservations are enforced using a series of reservations, extending a number of periods into the future as controlled by the **DEPTH** attribute of the [SRCFG](#) parameter.

The following examples demonstrate possible configurations specified with the **SRCFG** parameter.

**Example 3-108: Basic Business Hour Standing Reservation**

```

SRCFG[interactive] TASKCOUNT=6 RESOURCES=PROCS:1, MEM:512
SRCFG[interactive] PERIOD=DAY DAYS=MON, TUE, WED, THU, FRI
SRCFG[interactive] STARTTIME=9:00:00 ENDTIME=17:00:00
SRCFG[interactive] CLASSLIST=interactive

```

**i** When using the SRCFG parameter, attribute lists must be delimited using the comma (,), pipe (|), or colon (:) characters; they cannot be space delimited. For example, to specify a multi-class ACL, specify:

```
SRCFG[test] CLASSLIST=classA,classB
```

**i** Only one **STARTTIME** and one **ENDTIME** value can be specified per reservation. If varied start and end times are desired throughout the week, complementary standing reservations should be created. For example, to establish a reservation from 8:00 p.m. until 6:00 a.m. the next day during business days, two reservations should be created—one from 8:00 p.m. until midnight, and the other from midnight until 6:00 a.m. Jobs can run across reservation boundaries allowing these two reservations to function as a single reservation that spans the night. The following example demonstrates how to span a reservation across 2 days on the same nodes:

```

SRCFG[Sun] PERIOD=WEEK
SRCFG[Sun] STARTTIME=00:20:00:00 ENDTIME=01:00:00:00
SRCFG[Sun] HOSTLIST=node01,node02,node03

SRCFG[Mon] PERIOD=WEEK
SRCFG[Mon] STARTTIME=01:00:00:00 ENDTIME=01:06:00:00
SRCFG[Sun] HOSTLIST=node01,node02,node03

```

The preceding example fully specifies a reservation including the quantity of resources requested using the **TASKCOUNT** and **RESOURCES** attributes. In all cases, resources are allocated to a reservation in units called tasks where a task is a collection of resources that must be allocated together on a single node. The **TASKCOUNT** attribute specifies the number of these tasks that should be reserved by the reservation. In conjunction with this attribute, the **RESOURCES** attribute defines the reservation task by indicating what resources must be included in each task. In this case, the scheduler must locate and reserve 1 processor and 512 MB of memory together on the same node for each task requested.

As mentioned previously, a standing reservation reserves resources over a given time frame. The **PERIOD** attribute may be set to a value of **DAY**, **WEEK**, or **INFINITY** to indicate the period over which this reservation should recur. If not specified, a standing reservation recurs on a daily basis. If a standing reservation is configured to recur daily, the attribute **DAYS** may be specified to indicate which days of the week the reservation should exist. This attribute takes a comma-delimited list of days where each day is specified as the first three letters of the day in all capital letters: **MON** or **FRI**. The preceding example specifies that this reservation is periodic on a daily basis and should only exist on business days.

The time of day during which the requested tasks are to be reserved is specified using the **STARTTIME** and **ENDTIME** attributes. These attributes are specified in standard military time HH:MM:SS format and both **STARTTIME** and **ENDTIME** specification is optional defaulting to midnight at the beginning and end of

the day respectively. In the preceding example, resources are reserved from 9:00 a.m. until 5:00 p.m. on business days.

The final aspect of any reservation is the access control list indicating who or what can use the reserved resources. In the preceding example, the **CLASSLIST** attribute is used to indicate that jobs requesting the class "interactive" should be allowed to use this reservation.

## Specifying Reservation Resources

In most cases, only a small subset of standing reservation attributes must be specified in any given case. For example, by default, **RESOURCES** is set to `PROCS=-1` which indicates that each task should reserve all of the processors on the node on which it is located. This, in essence, creates a one task equals one node mapping. In many cases, particularly on uniprocessor systems, this default behavior may be easiest to work with. However, in SMP environments, the **RESOURCES** attribute provides a powerful means of specifying an exact, multi-dimensional resource set.

**i** An examination of the parameters documentation shows that the default value of **PERIOD** is *DAYS*. Thus, specifying this parameter in the preceding above was unnecessary. It was used only to introduce this parameter and indicate that other options exist beyond daily standing reservations.

### Example 3-109: Host Constrained Standing Reservation

Although the first example did specify a quantity of resources to reserve, it did not specify where the needed tasks were to be located. If this information is not specified, Moab attempts to locate the needed resources anywhere it can find them. The Example 1 reservation essentially discovers hosts where the needed resources can be found. If the **SPACEFLEX** reservation flag is set, then the reservation continues to float to the best hosts over the life of the reservation. Otherwise, it will be locked to the initial set of allocated hosts.

If a site wanted to constrain a reservation to a subset of available resources, this could be accomplished using the **HOSTLIST** attribute. The **HOSTLIST** attribute is specified as a comma-separated list of hostnames and constrains the scheduler to only select tasks from the specified list. This attribute can exactly specify hosts or specify them using host regular expressions. The following example demonstrates a possible use of the **HOSTLIST** attribute:

```
SRCFG[interactive] DAYS=MON,TUE,WED,THU,FRI
SRCFG[interactive] PERIOD=DAY
SRCFG[interactive] STARTTIME=10:00:00 ENDTIME=15:00:00
SRCFG[interactive] RESOURCES=PROCS:2, MEM:256
SRCFG[interactive] HOSTLIST=node001,node002,node005,node020
SRCFG[interactive] TASKCOUNT=6
SRCFG[interactive] CLASSLIST=interactive
```

*Note that the **HOSTLIST** attribute specifies a non-contiguous list of hosts. Any combination of hosts may be specified and hosts may be specified in any order. In this example, the **TASKCOUNT** attribute is also specified. These two attributes both apply constraints on the scheduler with **HOSTLIST** specifying where the tasks can be located and **TASKCOUNT** indicating how many total tasks may be allocated. In this example, six tasks are requested but only four hosts are specified. To handle this, if adequate resources are available, the scheduler may attempt to allocate more than one task per host. For example, assume that each host is a quad-processor system with 1 GB of memory. In such a case, the scheduler could allocate up to two tasks per host and even satisfy the **TASKCOUNT** constraint without using all of the hosts in the hostlist.*

**i** It is important to note that even if there is a one to one mapping between the value of **TASKCOUNT** and the number of hosts in **HOSTLIST**, the scheduler will not necessarily place one task on each host. If, for example, node001 and node002 were 8 processor SMP hosts with 1 GB of memory, the scheduler could locate up to four tasks on each of these hosts fully satisfying the reservation taskcount without even partially using the remaining hosts. (Moab will place tasks on hosts according to the policy specified with the [NODEALLOCATIONPOLICY](#) parameter.) If the hostlist provides more resources than what is required by the reservation as specified via **TASKCOUNT**, the scheduler will simply select the needed resources within the set of hosts listed.

## Enforcing Policies Via Multiple Reservations

Single reservations enable multiple capabilities. Combinations of reservations can further extend a site's capabilities to impose specific policies.

### Example 3-110: Reservation Stacking

If **HOSTLIST** is specified but **TASKCOUNT** is not, the scheduler will pack as many tasks as possible onto all of the listed hosts. For example, assume the site added a second standing reservation named *debug* to its configuration that reserved resources for use by certain members of its staff using the following configuration:

```

SRCFG[interactive] DAYS=MON, TUE, WED, THU, FRI
SRCFG[interactive] PERIOD=DAY
SRCFG[interactive] STARTTIME=10:00:00 ENDTIME=15:00:00
SRCFG[interactive] RESOURCES=PROCS:2, MEM:256
SRCFG[interactive] HOSTLIST=node001, node002, node005, node020
SRCFG[interactive] TASKCOUNT=6
SRCFG[interactive] CLASSLIST=interactive
SRCFG[debug] HOSTLIST=node001, node002, node003, node004
SRCFG[debug] USERLIST=helpdesk
SRCFG[debug] GROUPLIST=operations, sysadmin
SRCFG[debug] PERIOD=INFINITY

```

The new standing reservation is quite simple. Since **RESOURCES** is not specified, it will allocate all processors on each host that is allocated. Since **TASKCOUNT** is not specified, it will allocate every host listed in **HOSTLIST**. Since **PERIOD** is set to *INFINITY*, the reservation is always in force and there is no need to specify **STARTTIME**, **ENDTIME**, or **DAYS**.

The standing reservation has two access parameters set using the attributes **USERLIST** and **GROUPLIST**. This configuration indicates that the reservation can be accessed if any one of the access lists specified is satisfied by the job. In essence, reservation access is logically OR'd allowing access if the requester meets any of the access constraints specified. In this example, jobs submitted by either user *helpdesk* or any member of the groups *operations* or *sysadmin* can use the reserved resources (See [ACL Modifiers](#)).

Unless [ACL Modifiers](#) are specified, access is granted to the logical *OR* of access lists specified within a standing reservation and granted to the logical *AND* of access lists across different standing reservations. A comparison of the standing reservations *interactive* and *debug* in the preceding example indicates that they both can allocate hosts *node001* and *node002*. If *node001* had both of these reservations in place simultaneously and a job attempted to access this host during business hours when standing reservation *interactive* was active. The job could only use the *doubly* reserved resources if it requests the run class *interactive* and it meets the constraints of reservation *debug*—that is, that it is submitted by user *helpdesk* or by a member of the group *operations* or *sysadmin*.

As a rule, the scheduler does not stack reservations unless it must. If adequate resources exist, it can allocate reserved resources side by side in a single SMP host rather than on top of each other. In the case of a 16 processor SMP host with two 8 processor standing reservations, 8 of the processors on this host will be allocated to the first reservation, and 8 to the next. Any configuration is possible. The 16 processor hosts can also have 4 processors reserved for user "John," 10 processors reserved for group "Staff," with the remaining 2 processors available for use by any job.

Stacking reservations is not usually required but some site administrators choose to do it to enforce elaborate policies. There is no problem with doing so as long as you can keep things straight. It really is not too difficult a concept; it just takes a little getting used to. See the [Reservation Overview](#) section for a more detailed description of reservation use and constraints.

As mentioned earlier, by default the scheduler enforces standing reservations by creating a number of reservations where the number created is controlled by the **DEPTH** attribute. Each night at midnight, the scheduler updates its periodic non-floating standing reservations. By default, **DEPTH** is set to 2, meaning when the scheduler starts up, it will create two 24-hour reservations covering a total of two days' worth of time—a reservation for today and one for tomorrow. For daily reservations, at midnight, the reservations roll, meaning today's reservation expires and is removed, tomorrow's reservation becomes today's, and the scheduler creates a new reservation for the next day.

With this model, the scheduler continues creating new reservations in the future as time moves forward. Each day, the needed resources are always reserved. At first, all appears automatic but the standing reservation **DEPTH** attribute is in fact an important aspect of reservation rolling, which helps address certain site specific environmental factors. This attribute remedies a situation that might occur when a job is submitted and cannot run immediately because the system is backlogged with jobs. In such a case, available resources may not exist for several days out and the scheduler must reserve these future resources for this job. With the default **DEPTH** setting of two, when midnight arrives, the scheduler attempts to roll its standing reservations but a problem arises in that the job has now allocated the resources needed for the standing reservation two days out. Moab cannot reserve the resources for the standing reservation because they are already claimed by the job. The standing reservation reserves what it can but because all needed resources are not available, the resulting reservation is now smaller than it should be, or is possibly even empty.

If a standing reservation is smaller than it should be, the scheduler will attempt to add resources each iteration until it is fully populated. However, in the case of this job, the job is not going to release its reserved resources until it completes and the standing reservation cannot claim them until this time. The **DEPTH** attribute allows a site to specify how deep into the future a standing reservation should reserve its resources allowing it to claim the resources first and prevent this problem. If a partial standing reservation is detected on a system, it may be an indication that the reservation's **DEPTH** attribute should be increased.

In Example 3, the **PERIOD** attribute is set to *INFINITY*. With this setting, a single, permanent standing reservation is created and the issues of resource contention do not exist. While this eliminates the contention issue, infinite length standing reservations cannot be made periodic.

#### *Example 3-111: Multiple ACL Types*

In most cases, access lists within a reservation are logically OR'd together to determine reservation access. However, exceptions to this rule can be specified by using the required ACL marker—the asterisk (\*). Any ACL marked with this symbol is required and a job is only allowed to use a reservation if it meets all required ACLs and at least one non-required ACL (if specified). A common use for this facility is in conjunction with the **TIMELIMIT** attribute. This attribute controls the length of time a job may use the

resources within a standing reservation. This access mechanism can be AND'd or OR'd to the cumulative set of all other access lists as specified by the required ACL marker. Consider the following example configuration:

```

SRCFG[special] TASKCOUNT=32
SRCFG[special] PERIOD=WEEK
SRCFG[special] STARTTIME=1:08:00:00
SRCFG[special] ENDTIME=5:17:00:00
SRCFG[special] NODEFEATURES=largememory
SRCFG[special] TIMELIMIT=1:00:00*
SRCFG[special] QOSLIST=high,low,special-
SRCFG[special] ACCCOUNTLIST=!projectX,!projectY

```

The above configuration requests 32 tasks which translate to 32 nodes. The **PERIOD** attribute makes this reservation periodic on a weekly basis while the attributes **STARTTIME** and **ENDTIME** specify the week offsets when this reservation is to start and end (Note that the specification format has changed to DD:HH:MM:SS.). In this case, the reservation starts on Monday at 8:00 a.m. and runs until Friday at 5:00 p.m. The reservation is enforced as a series of weekly reservations that only cover the specified time frame. The **NODEFEATURES** attribute indicates that each of the reserved nodes must have the node feature "largememory" configured.

As described earlier, **TIMELIMIT** indicates that jobs using this reservation can only use it for one hour. This means the job and the reservation can only overlap for one hour. Clearly jobs requiring an hour or less of wallclock time meet this constraint. However, a four-hour job that starts on Monday at 5:00 a.m. or a 12-hour job that starts on Friday at 4:00 p.m. also satisfies this constraint. Also, note the **TIMELIMIT** required ACL marker, \*; it is set indicating that jobs must not only meet the **TIMELIMIT** access constraint but must also meet one or more of the other access constraints. In this example, the job can use this reservation if it can use the access specified via **QOSLIST** or **ACCOUNTLIST**; that is, it is assigned a QoS of *high*, *low*, or *special*, or the submitter of the job has an account that satisfies the *!projectX* and *!projectY* criteria. See the [QoS Overview](#) for more info about QoS configuration and usage.

## Affinity

Reservation ACLs allow or deny access to reserved resources but they may be configured to also impact a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort—using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity. Note the '-' (hyphen or negative sign) following the *special* in the **QOSLIST** values. This special mark indicates that QoS *special* should be granted access to this reservation but should be assigned negative affinity. Thus, the **QOSLIST** attribute specifies that QoS *high* and *low* should be granted access with positive affinity (use the reservation first where possible) and QoS *special* granted access with negative affinity (use the reservation only when no other resources are available).

Affinity status is granted on a per access object basis rather than a per access list basis and always defaults to positive affinity. In addition to negative affinity, neutral affinity can also be specified using the equal sign (=) as in QOSLIST[0] normal= high debug= low-.

When a job matches multiple ACLs for a reservation, the final node affinity for the node, job, and reservation combination is based on the last matching ACL entry found in the configuration file.

For example, given the following reservation ACLs, a job matching both will receive a negative affinity:

```
SRCFG[res1] USERLIST=joe+ MAXTIME<=4:00:00-
```

With the following reservation ACLs, a job matching both will receive a positive affinity:

```
SRCFG[res1] MAXTIME<=4:00:00- USERLIST=joe+
```

### ACL Modifiers

ACL modifiers allow a site to change the default behavior of ACL processing. By default, a reservation can be accessed if one or more of its ACLs can be met by the requestor. This behavior can be changed using the "deny" or "required" ACL modifier, as in the following tables:

Not	
<b>Symbol:</b>	! (exclamation point)
<b>Description</b>	If attribute is met, the requestor is denied access regardless of any other satisfied ACLs.
<b>Example</b>	<pre>SRCFG[test] GROUPLIST=staff USERLIST=!steve</pre> <p><i>Allow access to all staff members other than <b>steve</b>.</i></p>

Required	
<b>Symbol:</b>	* (asterisk)
<b>Description</b>	All required ACLs must be satisfied for requestor access to be granted.
<b>Example</b>	<pre>SRCFG[test] QOSLIST=*high MAXTIME=*2:00:00</pre> <p><i>Only jobs in QoS <b>high</b> that request less than 2 hours of walltime are granted access.</i></p>

XOR	
<b>Symbol:</b>	^ (carat)
<b>Description</b>	All attributes of the type specified other than the ones listed in the ACL satisfy the ACL.

XOR	
<b>Example</b>	<pre>SRCFG[test] QOSLIST=^high</pre> <p><i>All jobs other than those requesting QoS <b>high</b> are granted access.</i></p>

CredLock	
<b>Symbol:</b>	& (ampersand)
<b>Description</b>	Matching jobs will be required to run on the resources reserved by this reservation. You can use this modifier on accounts, classes, groups, qualities of service, and users.
<b>Example</b>	<pre>SRCFG[test] USERLIST=&amp;john</pre> <p><i>All of user <b>john</b>'s jobs must run in this reservation.</i></p>

HPEnable (hard policy enable)	
<b>Symbol:</b>	~ (tilde)
<b>Description</b>	ACLs marked with this modifier are ignored during soft policy scheduling and are only considered for hard policy scheduling once all eligible soft policy jobs start.
<b>Example</b>	<pre>SRCFG[johnspace] USERLIST=john CLASSLIST=~debug</pre> <p><i>All of user <b>john</b>'s jobs are allowed to run in the reservation at any time. <b>Debug</b> jobs are also allowed to run in this reservation but are only considered after all of John's jobs are given an opportunity to start. User <b>john</b>'s jobs are considered before debug jobs regardless of job priority.</i></p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> If HPEnable and <b>Not</b> markers are used in conjunction, then specified credentials are <i>blocked-out</i> of the reservation during soft-policy scheduling.</p> </div>

Note the **ACCOUNTLIST** values in [Example 3-111](#) are preceded with an exclamation point, or NOT symbol. This indicates that all jobs with accounts other than **projectX** and **projectY** meet the account ACL. Note that if a !<X> value (!projectX) appears in an ACL line, that ACL is satisfied by any object not explicitly listed by a NOT entry. Also, if an object matches a NOT entry, the associated job is excluded from the reservation even if it meets other ACL requirements. For example, a QoS 3 job requesting account **projectX** is denied access to the reservation even though the job QoS matches the QoS ACL.

*Example 3-112: Binding Users to Reservations at Reservation Creation*

```
# create a 4 node reservation for john and bind all of john's jobs to that reservation
> mrsvctl -c -a user=&john -t 4
```

## Reservation Ownership

Reservation ownership allows a site to control who owns the reserved resources during the reservation time frame. Depending on needs, this ownership may be identical to, a subset of, or completely distinct from the reservation ACL. By default, reservation ownership implies resource accountability and resources not consumed by jobs are accounted against the reservation owner. In addition, ownership can also be associated with special privileges within the reservation.

Ownership is specified using the **OWNER** attribute in the format `<CREDTYPE>:<CREDID>`, as in **OWNER=USER:john**. To enable *john's* jobs to preempt other jobs using resources within the reservation, the **SRCFG** attribute **FLAG** should be set to **OWNERPREEMPT**. In the example below, the *jupiter* project chooses to share resources with the *saturn* project but only when it does not currently need them.

*Example 3-113: Limited Shared Access*

```
ACCOUNTCFG[jupiter] PRIORITY=10000
SRCFG[jupiter] HOSTLIST=node0[1-9]
SRCFG[jupiter] PERIOD=INFINITY
SRCFG[jupiter] ACCOUNTLIST=jupiter, saturn-
SRCFG[jupiter] OWNER=ACCT:jupiter
SRCFG[jupiter] FLAGS=OWNERPREEMPT
```

## Partitions

A reservation can be used in conjunction with a partition. Configuring a standing reservation on a partition allows constraints to be (indirectly) applied to a partition.

*Example 3-114: Time Constraints by Partition*

The following example places a 3-day wall-clock limit on two partitions and a 64 processor-hour limit on jobs running on partition *small*.

```
SRCFG[smallrsv] PARTITION=small MAXTIME=3:00:00:00 PSLIMIT<=230400 HOSTLIST=ALL
SRCFG[bigrsv] PARTITION=big MAXTIME=3:00:00:00 HOSTLIST=ALL
```

## Resource Allocation Behavior

As mentioned, standing reservations can operate in one of two modes, floating, or non-floating (essentially node-locked). A floating reservation is created when the flag **SPACEFLEX** is specified. If a reservation is non-floating, the scheduler allocates all resources specified by the **HOSTLIST** parameter regardless of node state, job load, or even the presence of other standing reservations. Moab interprets the request for a non-floating reservation as, "I want a reservation on these exact nodes, no matter what!"

If a reservation is configured to be floating, the scheduler takes a more relaxed stand, searching through all possible nodes to find resources meeting standing reservation constraints. Only *Idle*, *Running*, or *Busy* nodes are considered and further, only considered if no reservation conflict is detected. The reservation

attribute **ACCESS** modifies this behavior slightly and allows the reservation to allocate resources even if reservation conflicts exist.

**i** If a **TASKCOUNT** is specified with or without a **HOSTEXPRESSION**, Moab will, by default, only consider "up" nodes for allocation. To change this behavior, the reservation flag **IGNSTATE** can be specified as in the following example:

```
SRCFG[nettest] GROUPLIST=sysadm
SRCFG[nettest] FLAGS=IGNSTATE
SRCFG[nettest] HOSTLIST=node1 [3-8]
SRCFG[nettest] STARTTIME=9:00:00
SRCFG[nettest] ENDTIME=17:00:00
```

**i** Access to existing reservations can be controlled using the reservation flag **IGNRSV**.

Other standing reservation attributes not covered here include **PARTITION** and **CHARGEACCOUNT**. These parameters are described in some detail in the [parameters](#) documentation.

#### Example 3-115: Using Reservations to Guarantee Turnover

In some cases, it is desirable to make certain a portion of a cluster's resources are available within a specific time frame. The following example creates a floating reservation belonging to the *jupiter* account that guarantees 16 tasks for use by jobs requesting up to one hour.

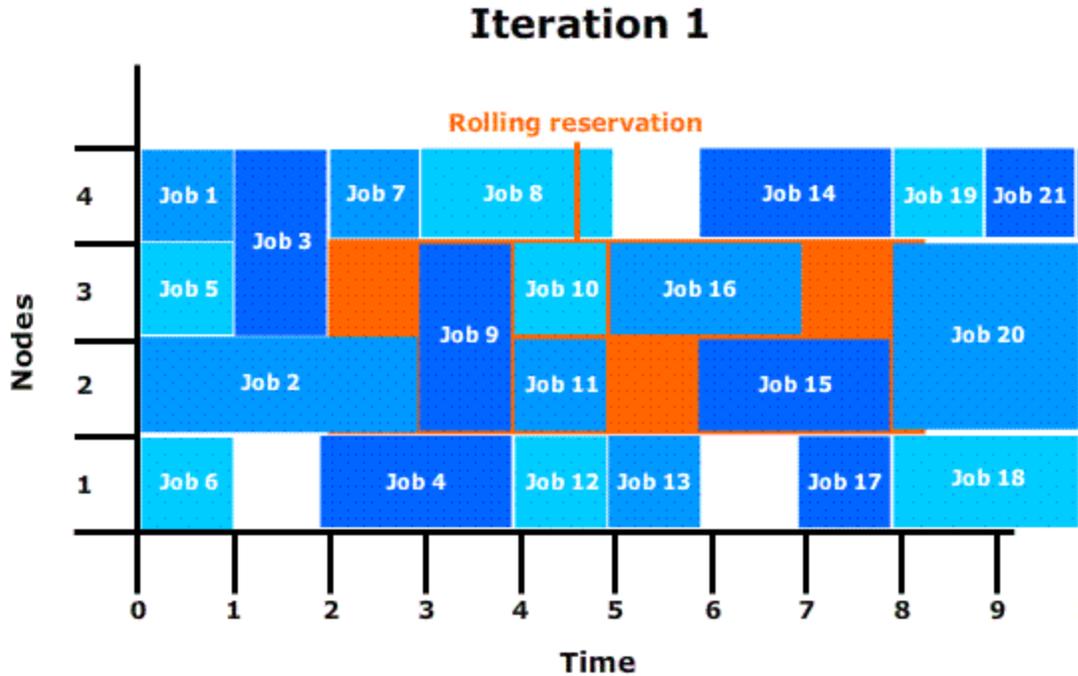
```
SRCFG[shortpool] OWNER=ACCT:jupiter
SRCFG[shortpool] FLAGS=SPACEFLEX
SRCFG[shortpool] MAXTIME=1:00:00
SRCFG[shortpool] TASKCOUNT=16
SRCFG[shortpool] STARTTIME=9:00:00
SRCFG[shortpool] ENDTIME=17:00:00
SRCFG[shortpool] DAYS=Mon,Tue,Wed,Thu,Fri
```

This reservation enables a capability similar to what was known in early Maui releases as "shortpool." The reservation covers every weekday from 9:00 a.m. to 5:00 p.m., reserving 16 tasks and allowing jobs to overlap the reservation for up to one hour. The **SPACEFLEX** flag indicates that the reservation may be dynamically modified--over time to re-locate to more optimal resources. In the case of a reservation with the **MAXTIME** ACL, this would include migrating to resources that are in use but that free up within the **MAXTIME** time frame. Additionally, because the **MAXTIME** ACL defaults to positive [affinity](#), any jobs that fit the ACL attempt to use available reserved resources first before looking elsewhere.

## Rolling Reservations

Rolling reservations are enabled using the [ROLLBACKOFFSET](#) attribute and can be used to allow users guaranteed access to resources, but the guaranteed access is limited to a time-window in the future. This functionality forces users to commit their resources in the future or lose access.

Image 3-5: Rolling reservation over 3 iterations



*Example 3-116: Rollback Reservations*

```

SRCFG[ajax] ROLLBACKOFFSET=24:00:00 TASKCOUNT=32
SRCFG[ajax] PERIOD=INFINITY ACCOUNTLIST=ajax
    
```

Adding an asterisk to the **ROLLBACKOFFSET** value pins rollback reservation start times when an idle reservation is created in the rollback reservation. For example: SRCFG[staff] ROLLBACKOFFSET=18:00:00\* PERIOD=INFINITY.

**Modifying Resources with Standing Reservations**

Moab can customize compute resources associated with a reservation during the life of the reservation. This can be done generally using the **TRIGGER** attribute, or it can be done for operating systems using the shortcut attribute **OS**. If set, Moab dynamically reprovisions allocated reservation nodes to the requested operating system as shown in the following example:

```

SRCFG[provision] PERIOD=DAY DAY=MON,WED,FRI STARTTIME=7:00:00 ENDTIME=10:00:00
SRCFG[provision] OS=rhel4 # provision nodes to use redhat during reservation, restore
when done
    
```

**Managing Administrative Reservations**

A default reservation with no ACL is termed an *administrative* reservation, but is occasionally referred to as a *system* reservation. It blocks access to all jobs because it possesses an empty access control list. It

is often useful when performing administrative tasks but cannot be used for enforcing resource usage policies.

Administrative reservations are created and managed using the [mrsvctl](#) command. With this command, all aspects of reservation time frame, resource selection, and access control can be dynamically modified. The [mdiag -r](#) command can be used to view configuration, state, allocated resource information as well as identify any potential problems with the reservation. The following table briefly summarizes commands used for common actions. More detailed information is available in the command summaries.

Action	Command
<b>create reservation</b>	<code>mrsvctl -c &lt;RSV_DESCRIPTION&gt;</code>
<b>list reservations</b>	<code>mrsvctl -l</code>
<b>release reservation</b>	<code>mrsvctl -r &lt;RSVID&gt;</code>
<b>modify reservation</b>	<code>mrsvctl -m &lt;ATTR&gt;=&lt;VAL&gt; &lt;RSVID&gt;</code>
<b>query reservation configuration</b>	<code>mdiag -r &lt;RSVID&gt;</code>
<b>display reservation hostlist</b>	<code>mrsvctl -q resources &lt;RSVID&gt;</code>

#### Related topics

- [SRCFG](#) (configure standing reservations)
- [RSVPROFILE](#) (create reservation profiles)

## Personal Reservations

- [Enabling Personal Reservation Management](#)
- [Reservation Accountability and Defaults](#)
  - [Reservation Allocation and Charging](#)
  - [Setting Reservation Default Attributes](#)
- [Reservation Limits](#)
- [Reservation and Job Binding](#)
  - [Constraining a job to only run in a particular reservation](#)
  - [Constraining a Reservation to Only Accept Certain Jobs](#)

By default, advance reservations are only available to scheduler administrators. While administrators may create and manage reservations to provide resource access to end-users, end-users cannot create, modify, or destroy these reservations. Moab extends the ability to manage reservations to end-users and

provides control facilities to keep these features manageable. Reservations created by end-users are called personal reservations or user reservations.

## Enabling Personal Reservation Management

User, or personal, reservations can be enabled on a per QoS basis by setting the [ENABLEUSERRSV](#) flag as in the following example:

```
QOSCFG[titan]    QFLAGS=ENABLEUSERRSV # allow 'titan' QoS jobs to create user
reservations
USERCFG[DEFAULT] QDEF=titan          # allow all users to access 'titan' QoS
...
```

If set, end-users are allowed to create, modify, cancel, and query reservations they own. As with jobs, users may associate a personal reservation with any QoS or account to which they have access. This is accomplished by specifying per reservation accountable credentials as in the following example:

```
> mrsvctl -c -S AQOS=titan -h node01 -d 1:00:00 -s 1:30:00
Note: reservation test.126 created
```

As in the preceding example, a non-administrator user who wants to create a reservation must *ALWAYS* specify an accountable QoS with the [mrsvctl -S](#) flag. This specified QoS must have the **ENABLEUSERRSV** flag. By default, a personal reservation is created with an ACL of only the user who created it.

*Example 3-117: Allow All Users in Engineering Group to Create Personal Reservations*

```
QOSCFG[rsv]      QFLAGS=ENABLEUSERRSV # allow 'rsv' QoS jobs to create user
reservations
GROUPCFG[sales] QDEF=rsv              # allow all users in group sales to access 'rsv'
QoS
...
```

*Example 3-118: Allow Specific Users to Create Personal Reservations*

```
# special qos has higher job priority and ability to create user reservations
QOSCFG[special] QFLAGS=ENABLEUSERRSV
QOSCFG[special] PRIORITY=1000
# allow betty and steve to use the special qos
USERCFG[betty]  QDEF=special
USERCFG[steve]  QLIST=fast,special,basic QDEF=rsv
...
```

## Reservation Accountability

Personal reservations must be configured with a set of accountable credentials. These credentials (user, group, account, and so forth) indicate who is responsible for the resources dedicated by the reservation. If resources are dedicated by a reservation but not consumed by a job, these resources can be charged against the specified accountable credentials. Administrators are allowed to create reservations and specify any accountable credentials for that reservation. While end-users can also be allowed to create and otherwise modify personal reservations, they are only allowed to create reservations with accountable credentials to which they have access. Further, while administrators may manage any reservation, end-users may only control reservations they own.

Like jobs, reservation accountable credentials specify which credentials are charged for reservation usage and what policies are enforced as far as usage limits and allocation management is concerned. (See the [mrsvctl](#) command documentation for more information on setting personal reservation credentials.) While similar to jobs, personal reservations do have a separate set of usage limits and different allocation charging policies.

## Setting Reservation Default Attributes

Organizations can use [reservation profiles](#) to set default attributes for personal reservations. These attributes can include reservation aspects such as management policies, charging credentials, ACLs, host constraints, and time frame settings.

## Reservation Limits

Allowing end-users the ability to create advance reservations can lead to potentially unfair and unproductive resource usage. This results from the fact that by default, there is nothing to prevent a user from reserving all resources in a given system or reserving resources during time slots that would greatly impede the scheduler's ability to schedule jobs efficiently. Because of this, it is highly advised that sites initially place either usage or allocation based constraints on the use of personal reservations. This can be achieved using Moab Accounting Manager (see the Moab Accounting Manager Administrator Guide).

## Reservation and Job Binding

Moab allows job-to-reservation binding to be configured at an administrator or end-user level. This binding constrains how job to reservation mapping is allowed.

### Constraining a job to only run in a particular reservation

Jobs may be bound to a particular reservation at submit time (using the RM extension [ADVRES](#)) or dynamically using the [mjobctl](#) command (See [Job to Reservation Mapping](#)). In either case, once bound to a reservation, a job may only run in that reservation even if other resources may be found outside of that reservation. The [mjobctl](#) command may also be used to dynamically release a job from reservation binding.

*Example 3-119: Bind job to reservation*

```
> mjobctl -m flags+=advres:grid.3 job1352
```

*Example 3-120: Release job from reservation binding*

```
> mjobctl -m flags-=advres job1352
```

### Constraining a Reservation to Only Accept Certain Jobs

Binding a job to a reservation is independent of binding a reservation to a job. For example, a reservation may be created for user "steve." User "steve" may then submit a number of jobs including one that is bound to that reservation using the **ADVRES** attribute. However, this binding simply forces that one job to use the reservation, it does not prevent the reservation from accepting other jobs

submitted by user "steve." To prevent these other jobs from using the reserved resources, reservation to job binding must occur. This binding is accomplished by specifying either general job binding or specific job binding.

General job binding is the most flexible form of binding. Using the [BYNAME](#) attribute, a reservation may be created that only accepts jobs specifically bound to it.

Specific job binding is more constraining. This form of binding causes the reservation to only accept specific jobs, regardless of other job attributes and is set using the **JOB** reservation ACL.

*Example 3-121: Configure a reservation to accept only jobs that are bound to it*

```
> mrsvctl -m flags+=byname grid.3
```

*Example 3-122: Remove general reservation to job binding*

```
> mrsvctl -m flags-=byname grid.3
```

*Example 3-123: Configure a reservation to accept a specific job*

```
> mrsvctl -m -a JOB=3456 grid.3
```

*Example 3-124: Remove a specific reservation to job binding*

```
> mrsvctl -m -a JOB=3456 grid.3 --flags=unset
```

## Partitions

- [Partition Overview](#)
- [Defining Partitions](#)
- [Managing Partition Access](#)
- [Requesting Partitions](#)
- [Per-Partition Settings](#)
- [Miscellaneous Partition Issues](#)

### Partition Overview

Partitions are a logical construct that divide available resources. Any single resource (compute node) may only belong to a single partition. Often, natural hardware or resource manager bounds delimit partitions such as in the case of disjoint networks and diverse processor configurations within a cluster. For example, a cluster may consist of 256 nodes containing four 64 port switches. This cluster may receive excellent interprocess communication speeds for parallel job tasks located within the same switch but sub-stellar performance for tasks that span switches. To handle this, the site may choose to create four partitions, allowing jobs to run within any of the four partitions but not span them.

While partitions do have value, it is important to note that within Moab, the [standing reservation](#) facility provides significantly improved flexibility and should be used in the vast majority of politically

motivated cases where partitions may be required under other resource management systems. Standing reservations provide time flexibility, improved access control features, and more extended resource specification options. Also, another Moab facility called [Node Sets](#) allows intelligent aggregation of resources to improve per job node allocation decisions. In cases where system partitioning is considered for such reasons, node sets may be able to provide a better solution.

Still, one key advantage of partitions over standing reservations and node sets is the ability to specify partition specific policies, limits, priorities, and scheduling algorithms although this feature is rarely required. An example of this need may be a cluster consisting of 48 nodes owned by the Astronomy Department and 16 nodes owned by the Mathematics Department. Each department may be willing to allow sharing of resources but wants to specify how their partition will be used. As mentioned, many of Moab's scheduling policies may be specified on a per partition basis allowing each department to control the scheduling goals within their partition.

The partition associated with each node should be specified as indicated in the [Node Location](#) section. With this done, partition access lists may be specified on a per job or per QoS basis to constrain which resources a job may have access to. (See the [QoS Overview](#) for more information.) By default, QoSs and jobs allow global partition access. Note that by default, a job may only use resources within a single partition.

If no partition is specified, Moab creates one partition per resource manager into which all resources corresponding to that resource manager are placed. (This partition is given the same name as the resource manager.)

**i** A partition may not span multiple resource managers. In addition to these resource manager partitions, a pseudo-partition named "[ALL]" is created that contains the aggregate resources of all partitions.

**i** While the resource manager partitions are real partitions containing resources not explicitly assigned to other partitions, the "[ALL]" partition is only a convenience object and is not a real partition; thus it cannot be requested by jobs or included in configuration ACLs.

## Defining Partitions

Node to partition mappings can be established directly using the [NODECFG](#) parameter or indirectly using the [FEATUREPARTITIONHEADER](#) parameter. If using direct mapping, this is accomplished as shown in the example that follows.

```
NODECFG[node001] PARTITION=astronomy
NODECFG[node002] PARTITION=astronomy
...
NODECFG[node049] PARTITION=math
...
```

**i** By default, Moab creates two partitions, "DEFAULT" and "[ALL]." These are used internally, and consume spots in the 31-partition maximum defined in the [MMAX\\_PAR](#) parameter. If more partitions are needed, you can adjust the maximum partition count. See [Adjusting Default Limits](#) for information on increasing the maximum number of partitions.

## Managing Partition Access

Partition access can be constrained by credential ACLs and by limits based on job resource requirements.

### Credential Based Access

Determining who can use which partition is specified using the **\*CFG** parameters ([USERCFG](#), [GROUPCFG](#), [ACCOUNTCFG](#), [QOSCFG](#), [CLASSCFG](#), and [SYSCFG](#)). These parameters allow you to select a partition access list on a credential or system wide basis using the **PLIST** attribute. By default, the access associated with any given job is the logical OR of all partition access lists assigned to the job's credentials.

For example, assume a site with two partitions, *general*, and *test*. The site management would like everybody to use the *general* partition by default. However, one user, Steve, needs to perform the majority of his work on the test partition. Two special groups, staff and management will also need access to use the test partition from time to time but will perform most of their work in the general partition. The following example configuration enables the needed user and group access and defaults for this site:

```
SYSCFG [base]      PLIST=general:test
USERCFG [DEFAULT] PLIST=general
USERCFG [steve]    PLIST=general:test
GROUPCFG [staff]   PLIST=general:test
GROUPCFG [mgmt]    PLIST=general:test
```

While using a logical OR approach allows sites to add access to certain jobs, some sites prefer to work the other way around. In these cases, access is granted by default and certain credentials are then restricted from accessing various partitions. To use this model, a system partition list must be specified as in the following example:

```
SYSCFG [base]      PLIST=general,test&
USERCFG [demo]     PLIST=test&
GROUPCFG [staff]   PLIST=general&
```

In the preceding example, note the ampersand (&). This character, which can be located anywhere in the **PLIST** line, indicates that the specified partition list should be logically AND'd with other partition access lists. In this case, the configuration limits jobs from user *demo* to running in partition *test* and jobs from group *staff* to running in partition *general*. All other jobs are allowed to run in either partition.

**i** When using AND-based partition access lists, the base system access list must be specified with **SYSCFG**.

### Per Job Resource Limits

Access to partitions can be constrained based on the resources requested on a per job basis with limits on both minimum and maximum resources requested. All limits are specified using [PARCFG](#). See [Usage Limits](#) for more information on the available limits.

```
PARCFG [amd]      MAX.PROC=16
PARCFG [pIII]     MAX.WCLIMIT=12:00:00 MIN.PROC=4
PARCFG [aix]      MIN.NODE=12
```

## Requesting Partitions

Users may request to use any partition they have access to on a per job basis. This is accomplished using the resource manager extensions since most native batch systems do not support the partition concept. For example, on a [TORQUE](#) system, a job submitted by a member of the group *staff* could request that the job run in the *test* partition by adding the line `-l partition=test` to the `qsub` command line. See the [resource manager extension overview](#) for more information on configuring and using resource manager extensions.

## Per-Partition Settings

The following settings can be specified on a per-partition basis using the [PARCFG](#) parameter:

Setting	Description
<b>GMETRIC</b>	<p>Specifies a <a href="#">generic metric</a> to apply to the partition. It is configured like a Moab parameter, with the <code>gmetric</code> name inside square brackets. Specify multiple gmetrics by separating each configuration with a space. For example:</p> <pre>PARCFG[par1] GMETRIC[GM1]=20 GMETRIC[GM2]=10</pre> <p><i>Partition par1 has a GM1 metric of 20 and a GM2 metric of 10.</i></p>
<b>JOBNODEMATCHPOLICY</b>	Specifies the <a href="#">JOBNODEMATCHPOLICY</a> to be applied to jobs that run in the specified partition.
<b>NODEACCESSPOLICY</b>	Specifies the <a href="#">NODEACCESSPOLICY</a> to be applied to jobs that run in the specified partition.
<b>NODEALLOCATIONPOLICY</b>	Specifies the <a href="#">NODEALLOCATIONPOLICY</a> to be applied to jobs that run in the specified partition.
<b>USETTC</b>	Specifies whether <a href="#">TTC</a> specified at submission should be used and displayed by the scheduler.
<b>VMCREATEDURATION</b>	Specifies the maximum amount of time VM creation can take before Moab considers it a failure (in [HH[:MM[:SS]]). If no value is set, there is no maximum limit.
<b>VMDELETEDURATION</b>	Specifies the maximum amount of time VM deletion can take before Moab considers it a failure (in [HH[:MM[:SS]]). If no value is set, there is no maximum limit.
<b>VMMIGRATEDURATION</b>	Specifies the maximum amount of time VM migration can take before Moab considers it a failure (in [HH[:MM[:SS]]). If no value is set, there is no maximum limit.

## Miscellaneous Partition Issues

A brief caution: Use of partitions has been quite limited in recent years as other, more effective approaches are selected for site scheduling policies. Consequently, some aspects of partitions have received only minor testing. Still, note that partitions are fully supported and any problem found will be rectified.

### Related topics

- [Standing Reservations](#)
- [Node Sets](#)
- [FEATUREPARTITIONHEADER](#) parameter
- [PARCFG](#) parameter

## Quality of Service (QoS) Facilities

This section describes how to do the following:

- Allow key projects access to special services (such as preemption, resource dedication, and advance reservations).
- Provide access to special resources by requested QoS.
- Enable special treatment within priority and fairshare facilities by requested QoS.
- Provide exemptions to usage limits and other policies by requested QoS.
- Specify delivered service and response time targets.
- Enable job deadline guarantees.
- Control the list of QoSs available to each user and job.
- Enable special charging rates based on requested or delivered QoS levels.
- Enable limits on the extent of use for each defined QoS.
- Monitor current and historical usage for each defined QoS.

It contains the following sub-sections:

- [QoS Overview](#)
- [QoS Enabled Privileges](#)
  - [Special Prioritization](#)
  - [Service Access and Constraints](#)
  - [Usage Limits and Overrides](#)
  - [Service Access Thresholds](#)
  - [Preemption Management](#)
- [Managing QoS Access](#)

- [Requesting QoS Services at Job Submission](#)
- [Restricting Access to Special Attributes](#)

## QoS Overview

Moab's QoS facility allows a site to give special treatment to various classes of jobs, users, groups, and so forth. Each QoS object can be thought of as a container of special privileges ranging from fairness policy exemptions, to special job prioritization, to special functionality access. Each QoS object also has an extensive access list of users, groups, and accounts that can access these privileges.

Sites can configure various QoSs each with its own set of priorities, policy exemptions, and special resource access settings. They can then configure user, group, account, and class access to these QoSs. A given job will have a default QoS and may have access to several additional QoSs. When the job is submitted, the submitter may request a specific QoS or just allow the default QoS to be used. Once a job is submitted, a user may adjust the QoS of the job at any time using the [setqos](#) command. The `setqos` command will only allow the user to modify the QoS of that user's jobs and only change the QoS to a QoS that this user has access to. Moab administrators may change the QoS of any job to any value.

Jobs can be granted access to QoS privileges if the QoS is listed in the system default configuration [QDEF](#) (QoS default) or [QLIST](#) (QoS access list), or if the QoS is specified in the [QDEF](#) or [QLIST](#) of a [user](#), [group](#), [account](#), or [class](#) associated with that job. Alternatively, a user may access QoS privileges if that user is listed in the QoSs [MEMBERULIST](#) attribute.

The [mdiag -q](#) command can be used to obtain information about the current QoS configuration including specified credential access.

## QoS Enabled Privileges

The privileges enabled via QoS settings may be broken into the following categories:

- [Special Prioritization on page 499](#)
- [Service Access and Constraints on page 500](#)
- [Usage Limits and Overrides on page 503](#)
- [Service Access Thresholds on page 504](#)
- [Preemption Management on page 504](#)

All privileges are managed via the [QOSCFG](#) parameter.

### Special Prioritization

Attribute name	Description
<b>FSTARGET</b>	Specifies QoS <a href="#">fairshare</a> target.
<b>FSWEIGHT</b>	Sets QoS fairshare weight offset affecting a job's <a href="#">fairshare</a> priority component.

Attribute name	Description
<b>PRIORITY</b>	Assigns priority to all jobs requesting particular QoS.
<b>QTTARGET</b>	Sets QoS queuetime target affecting a job's <a href="#">target</a> priority component and QoS delivered.
<b>QTWEIGHT</b>	Sets QoS queuetime weight offset affecting a job's <a href="#">service</a> priority component.
<b>XFTARGET</b>	Sets QoS XFactor target affecting a job's <a href="#">target</a> priority component and QoS delivered.
<b>XFWEIGHT</b>	Sets QoS XFactor weight offset affecting a job's <a href="#">service</a> priority component.

Example 3-125:

```
# assign priority for all qos geo jobs
QOSCFG[geo] PRIORITY=10000
```

## Service Access and Constraints

The QoS facility can be used to enable special services and to disable default services. These services are enabled/disabled by setting the QoS **QFLAGS** attribute.

Flag Name	Description
<b>DEADLINE</b>	Job may request an absolute or relative completion <a href="#">deadline</a> and Moab will reserve resources to meet that deadline. (An alternative priority based deadline behavior is discussed in the <a href="#">PRIORITY FACTORS</a> section.)
<b>DEDICATED</b>	Moab dedicates all resources of an allocated node to the job meaning that the job will not share a node's compute resources with any other job.
<b>ENABLEUSERRSV</b>	Allow user or personal <a href="#">reservations</a> to be created and managed.
<b>IGNALL</b>	Scheduler ignores all resource usage policies for jobs associated with this QoS.

Flag Name	Description
<b>JOBPRIOACCRUALPOLICY</b>	<p>Specifies how Moab should track the dynamic aspects of a job's priority. The two valid values are <i>ACCRUE</i> and <i>RESET</i>.</p> <ul style="list-style-type: none"> <li>• <i>ACCRUE</i> indicates that the job will accrue queue time based priority from the time it is submitted unless it violates any of the policies not specified in <a href="#">JOBPRIOEXCEPTIONS</a>.</li> <li>• <i>RESET</i> indicates that it will accrue priority from the time it is submitted unless it violates any of the <a href="#">JOBPRIOEXCEPTIONS</a>. However, with <i>RESET</i>, if the job does violate <a href="#">JOBPRIOEXCEPTIONS</a> then its queue time based priority will be reset to 0.</li> </ul> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p><b>i</b> <b>JOBPRIOACCRUALPOLICY</b> is a global parameter, but can be configured to work only in <b>QOSCFG</b>:</p> <pre style="border: 1px dashed #0070C0; border-radius: 5px; padding: 5px; display: inline-block;">QOSCFG[arrays] JOBPRIOACCRUALPOLICY=ACCRUE</pre> </div> <p>The following old <b>JOBPRIOACCRUALPOLICY</b> values have been deprecated and should be adjusted to the following values:</p> <ul style="list-style-type: none"> <li>• QUEUEPOLICY = ACCRUE and JOBPRIOEXCEPTIONS SOFTPOLICY,HARDPOLICY</li> <li>• QUEUEPOLICYRESET = RESET and JOBPRIOEXCEPTIONS SOFTPOLICY,HARDPOLICY</li> <li>• ALWAYS = ACCRUE and JOBPRIOEXCEPTIONS ALL</li> <li>• FULLPOLICY = ACCRUE and JOBPRIOEXCEPTIONS NONE</li> <li>• FULLPOLICYRESET = RESET and JOBPRIOEXCEPTIONS NONE</li> </ul>
<b>JOBPRIOEXCEPTIONS</b>	<p>Specifies exceptions for calculating a job's dynamic priority (QUEUE TIME, XFACTOR, TARGETQUEUE TIME). Valid values are a comma delimited list of any of the following: <i>DEFER</i>, <i>DEPENDS</i>, <i>SOFTPOLICY</i>, <i>HARDPOLICY</i>, <i>IDLEPOLICY</i>, <i>USERHOLD</i>, <i>BATCHHOLD</i>, and <i>SYSTEMHOLD</i> (<i>ALL</i> or <i>NONE</i> can also be specified on their own).</p> <p>Normally, when a job violates a policy, is placed on hold, or has an unsatisfied dependency, it will not accrue priority. Exceptions can be configured to allow a job to accrue priority in spite of any of these violations. With <i>DEPENDS</i> a job will increase in priority even if there exists an unsatisfied dependency. With <i>SOFTPOLICY</i>, <i>HARDPOLICY</i>, or <i>IDLEPOLICY</i> a job can accrue priority despite violating a specific limit. With <i>DEFER</i>, <i>USERHOLD</i>, <i>BATCHHOLD</i>, or <i>SYSTEMHOLD</i> a job can accrue priority despite being on hold.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p><b>i</b> <b>JOBPRIOEXCEPTIONS</b> is a global parameter, but can be configured to work only in <b>QOSCFG</b>:</p> <pre style="border: 1px dashed #0070C0; border-radius: 5px; padding: 5px; display: inline-block;">QOSCFG[arrays] JOBPRIOEXCEPTIONS=IDLEPOLICY</pre> </div>
<b>NOBF</b>	Job is not considered for backfill.

Flag Name	Description
<b>NORESERVATION</b>	Job should never reserve resources regardless of priority.
<b>NTR</b>	<p>Job is prioritized as next to run (NTR) and backfill is disabled to prevent other jobs from jumping in front of ones with the NTR flag.</p> <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"> <p><b>i</b> It is important to note that jobs marked with this flag should not be blocked. If they are, Moab will stop scheduling because if a job is marked with this flag, no other jobs will be run until the flagged NTR (Next to Run) job starts. Consider using the <b>PRIORITY</b> attribute of the <a href="#">QOSCFG[&lt;QOSID&gt;]</a> on page 1022 parameter instead, when possible. Or, as you may encounter a scheduling delay for NTR-flagged jobs to start, consider using the <a href="#">RESERVATIONDEPTH</a> and <a href="#">RESERVATIONQOSLIST</a> parameters to provide better scheduling flow. See <a href="#">Reservation Policies</a> on page 457 (especially the section on Assigning Per-QoS Reservation Creation Rules) for more information.</p> </div>
<b>PREEMPTCONFIG</b>	User jobs may specify options to alter how preemption impacts the job such as <a href="#">min-preempttime</a> .
<b>PREEMPTTEE</b>	Job may be preempted by higher priority <b>PREEMPTOR</b> jobs.
<b>PREEMPTFSV</b>	Job may be preempted by higher priority <b>PREEMPTOR</b> jobs if it exceeds its <a href="#">fairshare</a> target when started.
<b>PREEMPTOR</b>	Job may preempt lower priority <b>PREEMPTTEE</b> jobs.
<b>PREEMPTSPV</b>	Job may be preempted by higher priority <b>PREEMPTOR</b> jobs if it currently violates a <a href="#">soft usage policy</a> limit.
<b>PROVISION</b>	If the job cannot locate available resources with the needed OS or software, the scheduler may provision a number of nodes to meet the needed OS or software requirements.
<b>RESERVEALWAYS</b>	Job should create resource reservation regardless of job priority.
<b>RUNNOW</b>	<p>Boosts a job's system priority and makes the job a preemptor.</p> <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"> <p><b>i</b> RUNNOW overrides resource restrictions such as MAXJOB or MAXPROC.</p> </div>
<b>TRIGGER</b>	The job is able to directly specify triggers.

Flag Name	Description
<b>USERRESERVED[:&lt;RSVID&gt;]</b>	Job may only use resources within accessible reservations. If <RSVID> is specified, job may only use resources within the specified reservation.

Example 3-126: For lowprio QoS job, disable backfill and make job preemptible

```
QOSCFG[lowprio] QFLAGS=NOBF,PREEMPTEE
```

Example 3-127: Bind all jobs to chemistry reservation

```
QOSCFG[chem-b] QFLAGS=USERRESERVED:chemistry
```

### Other QoS Attributes

In addition to the flags, there are attributes that alter service access.

Attribute name	Description
<b>SYSPRIO</b>	Sets the system priority on jobs associated with this QoS.

Example 3-128: All jobs submitted under a QoS sample receive a system priority of 1

```
QOSCFG[sample] SYSPRIO=1
```

### Per QoS Required Reservations

If desired, jobs associated with a particular QoS can be locked into a reservation or reservation group using the **REQRID** attribute. For example, to force jobs using QoS *jasper* to only use the resources within the *failsafe* standing reservation, use the following:

```
QOSCFG[jasper] REQRID=failsafe
...
```

### Usage Limits and Overrides

All credentials, including QoS, allow specification of job usage limits as described in the [Basic Fairness Policies](#) overview. In such cases, jobs are constrained by the most limiting of all applicable policies. With QoSs, an override limit may also be specified and with this limit, jobs are constrained by the override, regardless of other limits specified. The following parameters can override the throttling policies from other credentials:

**OMAXJOB**, **OMAXNODE**, **OMAXPE**, **OMAXPROC**, **OMAXPS**, **OMAXJPROC**, **OMAXJPS**, **OMAXJWC**, and **OMAXJNODE**.

(See [Usage Limits/Throttling Policies Override Limits](#).)

Example 3-129:

```
# staff QoS should have a limit of 48 jobs, ignoring the user limit
USERCFG[DEFAULT]    MAXJOB=10
QOSCFG[staff]       OMAXJOB=48
```

### Service Access Thresholds

Jobs can be granted access to services such as [preemption](#) and [reservation creation](#), and they can be granted access to resource reservations. However, with QoS thresholds, this access can be made conditional on the current queue time and XFactor metrics of an idle job. The following table lists the available QoS service thresholds:

Threshold attribute	Description
<b>PREEMPTQTTRESHOLD</b>	A job with this QoS becomes a preemptor if the specified queue time threshold is reached.
<b>PREEMPTXFTHRESHOLD</b>	A job with this QoS becomes a preemptor if the specified XFactor threshold is reached.
<b>RSVQTTRESHOLD</b>	A job with this QoS can create a <a href="#">job reservation</a> to guarantee resource access if the specified queue time threshold is reached.
<b>RSVXFTHRESHOLD</b>	A job with this QoS can create a <a href="#">job reservation</a> to guarantee resource access if the specified XFactor threshold is reached.
<b>ACLQTTRESHOLD</b>	A job with this QoS can access reservations with a corresponding QoS ACL only if the specified queue time threshold is reached.
<b>ACLXFTHRESHOLD</b>	A job with this QoS can access reservations with a corresponding QoS ACL only if the specified XFactor threshold is reached.
<b>TRIGGERQTTRESHOLD</b>	If a job with this QoS fails to run before this threshold is reached, any failure triggers associated with this QoS will fire.

### Preemption Management

Job [preemption](#) facilities can be controlled on a per-QoS basis using the [PREEMPTTEE](#) and [PREEMPTOR](#) flags. Jobs that are preemptible can optionally be constrained to only be preempted in a particular manner by specifying the QoS **PREEMTPOLICY** attribute as in the following example:

```
QOSCFG[special] QFLAGS=PREEMPTTEE PREEMTPOLICY=CHECKPOINT
```

For preemption to be effective, a job must be marked as a preemptee and must be enabled for the requested preemption type. For example, if the [PREEMTPOLICY](#) is set to suspend, a potential target job must be both a preemptee and marked with the job flag **SUSPENDABLE**. (See [suspension](#) for more

information.) If the target job is not suspendable, it will be either requeued or canceled. Likewise, if the **PREEMTPOLICY** is set to *requeue*, the job will be requeued if it is marked restartable. Otherwise, it will be canceled.

The minimum time a job must run before being considered eligible for preemption can also be configured on a per-QoS basis using the **PREEMPTMINTIME** parameter, which is analogous to the [JOBPREEMPTMINACTIVETIME](#). Conversely, **PREEMPTMAXTIME** sets a threshold for which a job is no longer eligible for preemption; see [JOBPREEMPTMAXACTIVETIME](#) for analogous details.

The **PREEMPTTEES** attribute allows you to specify which QoSs that a job in a specific QoS is allowed to preempt. The **PREEMPTTEES** list is a comma-delimited list of QoS IDs. When a **PREEMPTTEES** attribute is specified, a job using that QoS can only preempt jobs using QoSs listed in the **PREEMPTTEES** list. In turn, those QoSs must be flagged as **PREEMPTTEE** as in the following example:

```
QOSCFG[a] QFLAGS=PREEMPTOR PREEMPTTEES=b, c
QOSCFG[b] QFLAGS=PREEMPTTEE
QOSCFG[c] QFLAGS=PREEMPTTEE
```

*In the example, jobs in the 'a' QoS can only preempt jobs in the b and c QoSs.*

## Managing QoS Access

### Specifying Credential Based QoS Access

You can define the privileges allowed within a QoS by using the **QOSCFG** parameter; however, in most cases access to the QoS is enabled via credential specific **\*CFG** parameters, specifically the [USERCFG](#), [GROUPCFG](#), [ACCOUNTCFG](#), and [CLASSCFG](#) parameters, which allow defining QoS access lists and QoS defaults. Specify credential specific QoS access by using the **QLIST** and/or **QDEF** attributes of the associated credential parameter.

### QoS Access via Logical OR

To enable QoS access, the **QLIST** and/or **QDEF** attributes of the appropriate user, group, account, or class/queue should be specified as in the following example:

```
# user john's jobs can access QOS geo, chem, or staff with geo as default
USERCFG[john] QDEF=geo QLIST=geo,chem,staff
# group system jobs can access the development qos
GROUPCFG[systems] QDEF=development
# class batch jobs can access the normal qos
CLASSCFG[batch] QDEF=normal
```

By default, jobs may request a QoS if access to that QoS is allowed by any of the job's credentials. (In the previous example, a job from user *john* submitted to the class *batch* could request QoSs *geo*, *chem*, *staff*, or *normal*).

### QoS Access via Logical AND

If desired, QoS access can be masked or logically AND'd if the QoS access list is specified with a terminating ampersand (&) as in the following example:

```
# user john's jobs can access QoS geo, chem, or staff with geo as default
USERCFG[john]      QDEF=geo  QLIST=geo,chem,staff
# group system jobs can access the development qos
GROUPCFG[systems] QDEF=development
# class batch jobs can access the normal qos
CLASSCFG[batch]   QDEF=normal
# class debug jobs can only access the development or lowpri QoSs regardless of other
credentials
CLASSCFG[debug]   QLIST=development,lowpri&
```

### Specifying QoS Based Access

QoS access may also be specified from within the QoS object using the QoS **MEMBERLIST** attribute as in the following example:

```
# define qos premiere and grant access to users steve and john
QOSCFG[premiere]  PRIORITY=1000  QFLAGS=PREEMPTOR  MEMBERLIST=steve,john
```

**i** By default, if a job requests a QoS that it cannot access, Moab places a hold on that job. The [QOSREJECTPOLICY](#) can be used to modify this behavior.

## Requesting QoS Services at Job Submission

By default, jobs inherit a default QoS based on the user, group, class, and account associated with the job. If a job has access to multiple QoS levels, the submitter can explicitly request a particular QoS using the [QoS](#) resource manager [extension](#) as in the following example:

```
> msub -l nodes=1,walltime=100,qos=special3 job.cmd
```

## Restricting Access to Special Attributes

By default, Moab allows all users access to special attributes such as [node access policy](#). By enabling the QoS facility **SPECATTRS**, the access to these policies can be restricted. For example, to enable the facility, in the moab.cfg file, specify `QOSCFG[DEFAULT] SPECATTRS=`. Then, to allow access to the special attributes, indicate which special attributes a specific QoS may access.

```
QOSCFG[DEFAULT] SPECATTRS=
QOSCFG[high]   SPECATTRS=NACCESSPOLICY
```

### Related topics

- [Credential Overview](#)
- [Allocation Management Overview](#)
- [Rollback Reservations](#)
- [Job Deadlines](#)
- [Using QoS preemption](#)

# Optimizing Scheduling Behavior – Backfill and Node Sets

- [Optimization Overview](#) on page 507
- [Backfill](#) on page 508
- [Node Set Overview](#) on page 514

## Optimization Overview

Moab optimizes cluster performance. Every policy, limit, and feature is designed to allow maximum scheduling flexibility while enforcing the required constraints. A driving responsibility of the scheduler is to do all in its power to maximize system use and to minimize job response time while honoring the policies that make up the site's mission goals.

However, as all jobs are not created equal, optimization must be abstracted slightly further to incorporate this fact. Cluster optimization must also focus on targeted cycle delivery. In the scientific HPC community, the true goal of a cluster is to maximize delivered research. For businesses and other organizations, the purposes may be slightly different, but all organizations agree on the simple tenet that the cluster should optimize the site's mission goals.

To obtain this goal, the scheduler has several levels of optimization it performs:

Level	Description
<b>Workload Ordering</b>	<a href="#">Prioritizing workload</a> and utilizing <a href="#">backfill</a>
<b>Intelligent Resource Allocation</b>	Selecting those resources that best meet the job's needs or best enable future jobs to run (see <a href="#">node allocation</a> )
<b>Maximizing Intra-Job Efficiency</b>	Selecting the type of nodes, collection of nodes, and proximity of nodes required to maximize job performance by minimizing both job compute and inter-process communication time (see <a href="#">node sets</a> and <a href="#">node allocation</a> )
<b>Job Pre-emption</b>	Preempting jobs to allow the most important jobs to receive the best response time (see <a href="#">pre-emption</a> )
<b>Utilizing Flexible Policies</b>	Using policies that minimize blocking and resource fragmentation while enforcing needed constraints (see <a href="#">soft throttling policies</a> and <a href="#">reservations</a> )

# Backfill

- [Backfill Overview](#)
- [Backfill Algorithms](#)
- [Configuring Backfill](#)

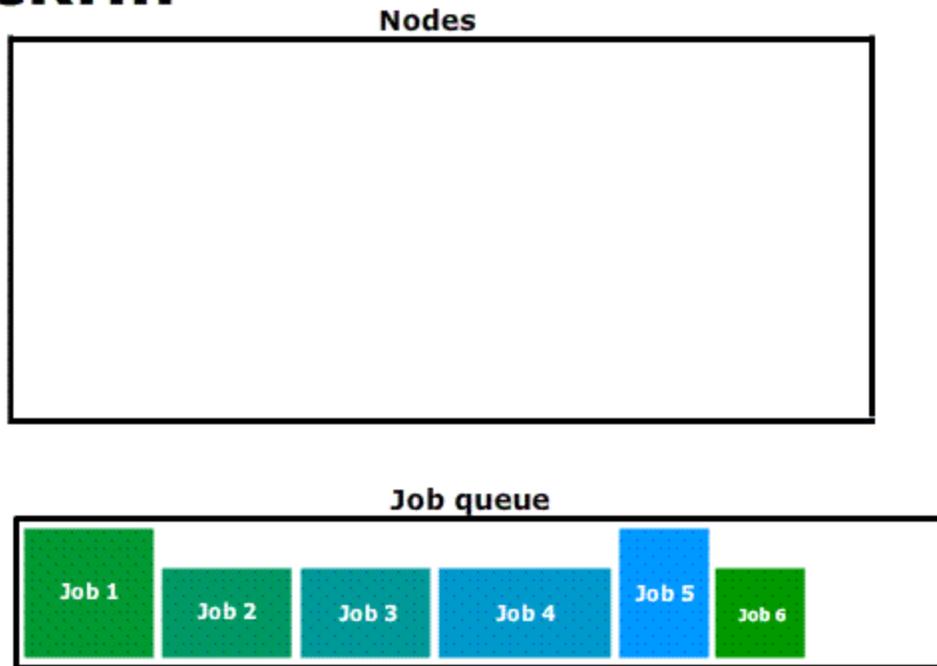
## Backfill Overview

Backfill is a scheduling optimization that allows a scheduler to make better use of available resources by running jobs out of order. When Moab schedules, it prioritizes the jobs in the queue according to a number of factors and then orders the jobs into a highest priority first (or priority FIFO) sorted list. It starts the jobs one by one stepping through the priority list until it reaches a job it cannot start. Because all jobs and reservations possess a start time and a wallclock limit, Moab can determine the completion time of all jobs in the queue. Consequently, Moab can also determine the earliest the needed resources will become available for the highest priority job to start.

Backfill operates based on this earliest job start information. Because Moab knows the earliest the highest priority job can start, and which resources it will need at that time, it can also determine which jobs can be started without delaying this job. Enabling backfill allows the scheduler to start other, lower-priority jobs so long as they do not delay the highest priority job. If backfill is enabled, Moab protects the highest priority job's start time by creating a job reservation to reserve the needed resources at the appropriate time. Moab then can start any job that will not interfere with this reservation.

Image 3-6: Scheduling with backfill

# Backfill



Backfill offers significant scheduler performance improvement. In a typical large system, enabling backfill increases system utilization by about 20% and improves turnaround time by an even greater amount. Because of the way it works, essentially filling in holes in node space, backfill tends to favor smaller and shorter running jobs more than larger and longer running ones. It is common to see over 90% of these small and short jobs backfilled. Consequently, sites will see marked improvement in the level of service delivered to the small, short jobs and moderate to little improvement for the larger, long ones.

With most algorithms and policies, there is a trade-off. Backfill is not an exception but the negative effects are minor. Because backfill locates jobs to run from throughout the idle job queue, it tends to diminish the influence of the job prioritization a site has chosen and thus may negate some desired workload steering attempts through this prioritization. Although by default the start time of the highest priority job is protected by a reservation, there is nothing to prevent the third priority job from starting early and possibly delaying the start of the second priority job. This issue is addressed along with its trade-offs [later](#) in this section.

Another problem is a little more subtle. Consider the following scenario involving a two-processor cluster. Job A has a four-hour wallclock limit and requires one processor. It started one hour ago (time zero) and will reach its wallclock limit in three more hours. Job B is the highest priority idle job and requires two processors for one hour. Job C is the next highest priority job and requires one processor for two hours. Moab examines the jobs and correctly determines that job A must finish in three hours and thus, the earliest job B can start is in three hours. Moab also determines that job C can start and finish in less than this amount of time. Consequently, Moab starts job C on the idle processor at time one. One hour later (time two), job A completes early. Apparently, the user overestimated the amount of time job A would need by a few hours. Since job B is now the highest priority job, it should be able to run. However, job C, a lower priority job was started an hour ago and the resources needed for job B are not available. Moab re-evaluates job B's reservation and determines that it can slide forward an hour. At time three, job B starts.

In review, backfill provided positive benefits. Job A successfully ran to completion. Job C was started immediately. Job B was able to start one hour sooner than its original target time, although, had backfill not been enabled, job B would have been able to run two hours earlier.

The scenario just described occurs quite frequently because user estimates for job duration are generally inaccurate. Job wallclock estimate accuracy, or wallclock accuracy, is defined as the ratio of wall time required to actually run the job divided by the wall time requested for the job. Wallclock accuracy varies from site to site but the site average is rarely better than 50%. Because the quality of the walltime estimate provided by the user is so low, job reservations for high priority jobs are often later than they need to be.

Although there do exist some minor drawbacks with backfill, its net performance impact on a site's workload is very positive. While a few of the highest priority jobs may get temporarily delayed, their position as highest priority was most likely accelerated by the fact that jobs in front of them were able to start earlier due to backfill. Studies have shown that only a very small number of jobs are truly delayed and when they are, it is only by a fraction of their total queue time. At the same time, many jobs are started significantly earlier than would have occurred without backfill.

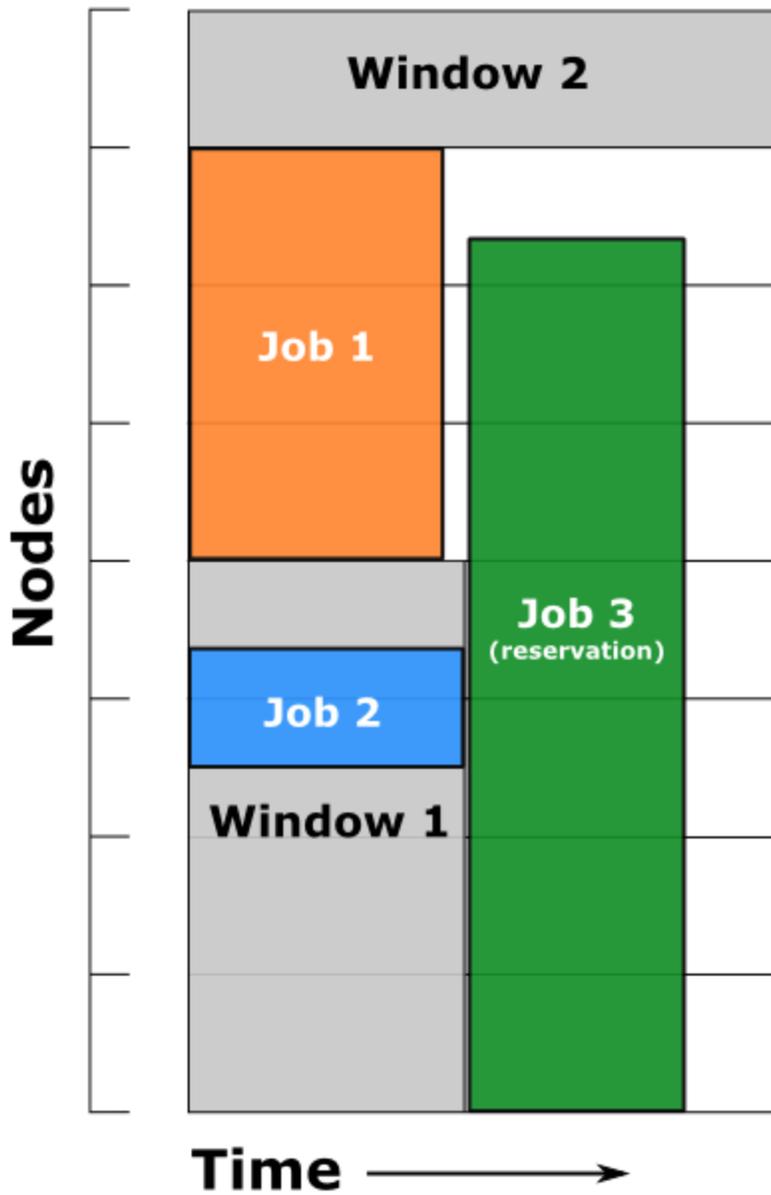
The following image demonstrates how Moab might schedule a queue using backfill.

## Backfill Algorithms

The algorithm behind Moab backfill scheduling is straightforward, although there are a number of issues and parameters that should be highlighted. First of all, Moab makes two backfill scheduling passes. For each pass, Moab selects a list of jobs that are eligible for backfill. On the first pass, only those jobs that meet the constraints of the soft [fairness throttling policies](#) are considered and scheduled. The second pass expands this list of jobs to include those that meet the hard (less constrained) fairness throttling policies.

The second important concept regarding Moab backfill is the concept of backfill windows. The figure below shows a simple batch environment containing two running jobs and a reservation for a third job. The present time is represented by the leftmost end of the box with the future moving to the right. The light gray boxes represent currently idle nodes that are eligible for backfill. For this example, let's assume that the space represented covers 8 nodes and a 2 hour time frame. To determine backfill windows, Moab analyzes the idle nodes essentially looking for largest node-time rectangles. It determines that there are two backfill windows. The first window, Window 1, consists of 4 nodes that are available for only one hour (because some of the nodes are blocked by the reservation for Job 3). The second window contains only one node but has no time limit because this node is not blocked by the reservation for Job 3. It is important to note that these backfill windows overlap.

Image 3-7: Backfillable nodes create backfill windows 1 and 2



Once the backfill windows have been determined, Moab begins to traverse them. The current behavior is to traverse these windows widest window first (most nodes to fewest nodes). As each backfill window is evaluated, Moab applies the backfill algorithm specified by the [BACKFILLPOLICY](#) parameter.

If the *FIRSTFIT* algorithm is applied, the following steps are taken:

1. The list of feasible backfill jobs is filtered, selecting only those that will actually fit in the current backfill window.
2. The first job is started.
3. While backfill jobs and idle resources remain, repeat step 1.

If *NONE* is set, the backfill policy is disabled.

Other backfill policies behave in a generally similar manner. The [parameters](#) documentation provides further details.

## Liberal versus Conservative Backfill

By default, Moab reserves only the highest priority job resulting in a liberal and aggressive backfill. This reservation guarantees that backfilled jobs will not delay the highest priority job, although they may delay other jobs. The parameter [RESERVATIONDEPTH](#) controls how conservative or liberal the backfill policy is. This parameter controls how deep down the queue priority reservations will be made. While increasing this parameter improves guarantees that priority jobs will not be bypassed, it reduces the freedom of the scheduler to backfill resulting in somewhat lower system utilization. The significance of the trade-offs should be evaluated on a site by site basis.

## Configuring Backfill

### Backfill Policies

Backfill is enabled in Moab by specifying the [BACKFILLPOLICY](#) parameter. By default, backfill is enabled in Moab using the *FIRSTFIT* algorithm. However, this parameter can also be set to *NONE* (disabled).

The number of reservations that protect the resources required by priority jobs can be controlled using [RESERVATIONDEPTH](#). This depth can be distributed across job QoS levels using [RESERVATIONQOSLIST](#).

### Backfill Chunking

In a batch environment saturated with serial jobs, serial jobs will, over time, dominate the resources available for backfill at the expense of other jobs. This is due to the time-dimension fragmentation associated with running serial jobs. For example, given an environment with an abundance of serial jobs, if a multi-processor job completes freeing processors, one of three things will happen:

1. The freed resources are allocated to another job requiring the same number of processors.
2. Additional jobs may complete at the same time allowing a larger job to allocate the aggregate resources.
3. The freed resources are allocated to one or more smaller jobs.

In environments where the scheduling iteration is much higher than the average time between completing jobs, case 3 occurs far more often than case 2, leading to smaller and smaller jobs populating the system over time.

To address this issue, the scheduler incorporates the concept of chunking. Chunking allows the scheduler to favor case 2 maintaining a more controlled balance between large and small jobs. The idea of chunking involves establishing a time-based threshold during which resources available for backfill are

aggregated. This threshold is set using the parameter [BFCHUNKDURATION](#). When resources are freed, they are made available only to jobs of a certain size (set using the parameter [BFCHUNKSIZE](#)) or larger. These resources remain protected from smaller jobs until either additional resources are freed up and a larger job can use the aggregate resources, or until the **BFCHUNKDURATION** threshold time expires.

**i** Backfill chunking is only activated when a job of size **BFCHUNKSIZE** or larger is blocked in backfill due to lack of resources.

It is important to note that the optimal settings for these parameters is very site-specific and will depend on the workload (including the average job turnaround time, job size, and mix of large to small jobs), cluster resources, and other scheduling environmental factors. Setting too restrictive values needlessly reduces utilization while settings that are too relaxed do not allowed the desired aggregation to occur.

**i** Backfill chunking is only enabled in conjunction with the *FIRSTFIT* backfill policy.

## Virtual Wallclock Time Scaling

In most environments, users submit jobs with rough estimations of the wallclock times. Within the HPC industry, a job typically runs for 40% of its specified wallclock time. Virtual Wallclock Time Scaling takes advantage of this fact to implement a form of optimistic backfilling. Jobs that are eligible for backfilling and not restricted by other policies are virtually scaled by the [BFVIRTUALWALLTIMESCALINGFACTOR](#) (assuming that the jobs finish before this new virtual wallclock limit). The scaled jobs are then compared to backfill windows to see if there is space and time for them to be scheduled. The scaled jobs are only scheduled if there is no possibility that it will conflict with a standing or administrator reservation. Conflicts with such reservations occur if the virtual wallclock time overlaps a reservation, or if the original non-virtual wallclock time overlaps a standing or administrator reservation. Jobs that can fit into an available backfill window without having their walltime scaled are backfilled "as-is" (meaning, without virtually scaling the original walltime).

**i** Virtual Wallclock Time Scaling is only enabled when the [BFVIRTUALWALLTIMESCALINGFACTOR](#) parameter is defined.

If a virtually-scaled job fits into a window, and is backfilled, it will run until completion or until it comes within one scheduling iteration ([RMPOLLINTERVAL](#) defines the exact time of an iteration) of the virtual wallclock time expiration. In the latter case the job's wallclock time is restored to its original time and Moab checks and resolves conflicts caused by this "expansion." Conflicts may occur when the backfilled job is restored to its full duration resulting in reservation overlap. The [BFVIRTUALWALLTIMECONFLICTPOLICY](#) parameter controls how Moab handles these conflicts.

If the **BFVIRTUALWALLTIMECONFLICTPOLICY** parameter is set to *NONE* or is not specified, the overlapped job reservations are rescheduled.

### Related topics

- [BACKFILLDEPTH](#) Parameter
- [BACKFILLPOLICY](#) Parameter

- [BFMINVIRTUALWALLTIME](#)
- [Reservation Policy Overview](#)

## Node Set Overview

- [Node Set Usage Overview](#)
- [Node Set Configuration](#)
  - [Node Set Policy](#)
  - [Node Set Attribute](#)
  - [Node Set Constraint Handling](#)
  - [Node Set List](#)
  - [Node Set Tolerance](#)
  - [Node Set Priority](#)
  - [NODESETPLUS](#)
  - [Nested Node Sets](#)
- [Requesting Node Sets for Job Submission](#)
- [Configuring Node Sets for Classes](#)

### Node Set Usage Overview

While backfill improves the scheduler's performance, this is only half the battle. The efficiency of a cluster, in terms of actual work accomplished, is a function of both scheduling performance and individual job efficiency. In many clusters, job efficiency can vary from node to node as well as with the node mix allocated. Most parallel jobs written in popular languages such as MPI or PVM do not internally load balance their workload and thus run only as fast as the slowest node allocated. Consequently, these jobs run most effectively on homogeneous sets of nodes. However, while many clusters start out as homogeneous, they quickly evolve as new generations of compute nodes are integrated into the system. Research has shown that this integration, while improving scheduling performance due to increased scheduler selection, can actually decrease average job efficiency.

A feature called node sets allows jobs to request sets of common resources without specifying exactly what resources are required. Node set policy can be specified globally or on a per-job basis. In addition to their use in forcing jobs onto homogeneous nodes, these policies may also be used to guide jobs to one or more types of nodes on which a particular job performs best, similar to job preferences available in other systems. For example, an I/O intensive job may run best on a certain range of processor speeds, running slower on slower nodes, while wasting cycles on faster nodes. A job may specify `ANYOF:FEATURE:bigmem,fastos` to request nodes with the `bigmem` or `fastos` feature. Alternatively, if a simple feature-homogeneous node set is desired, `ONEOF:FEATURE` may be specified. On the other hand, a job may request a feature based node set with the configuration `ONEOF:FEATURE:bigmem,fastos`, in which case Moab will first attempt to locate adequate nodes where all nodes contain the `bigmem` feature. If such a set cannot be found, Moab will look for sets of

nodes containing the other specified features. In highly heterogeneous clusters, the use of node sets improves job throughput by 10 to 15%.

Node sets can be requested on a system wide or per job basis. System wide configuration is accomplished via the **NODESET\*** parameters while per job specification occurs via the [resource manager extensions](#).

 The GLOBAL node is included in all feature node sets.

When creating node sets, you have the option of using a fixed configuration or of creating node sets dynamically (by using the `msub` command). This topic explains how to set up both node set use cases.

## Node Set Configuration Examples

Global node sets are defined using the [NODESETPOLICY](#), [NODESETATTRIBUTE](#), [NODESETLIST](#), and [NODESETISOPTIONAL](#) parameters. As stated before, you can create node sets dynamically (see [Dynamic example on page 517](#)) or with a fixed configuration (see [Fixed configuration example on page 515](#)). The use of these parameters can be best highlighted with two examples.

### Fixed configuration example

In this example, a large site possesses a Myrinet based interconnect and wishes to, whenever possible, allocate nodes within Myrinet switch boundaries. To accomplish this, they could assign node attributes to each node indicating which switch it was associated with (`switchA`, `switchB`, and so forth) and then use the following system wide node set configuration:

```
NODESETPOLICY      ONEOF
NODESETATTRIBUTE   FEATURE
NODESETISOPTIONAL TRUE
NODESETLIST        switchA,switchB,switchC,switchD
...
```

### Node Set Policy

In the preceding example, the [NODESETPOLICY](#) parameter is set to the policy **ONEOF** and tells Moab to allocate nodes within a single attribute set. Other node set policies are listed in the following table:

Policy	Description
<b>ANYOF</b>	Select resources from all sets contained in node set list. The job could span multiple node sets.
<b>FIRSTOF</b>	Select resources from first set to match specified constraints.
<b>ONEOF</b>	Select a single set that contains adequate resources to support job.

### Node Set Attribute

The example's [NODESETATTRIBUTE](#) parameter is set to **FEATURE**, specifying that the node sets are to be constructed along node feature boundaries.

You could also set the **NODESETATTRIBUTE** to *VARATTR*, specifying that node sets are to be constructed according to VARATTR values on the job.

### Node Set Constraint Handling

The next parameter, **NODESETISOPTIONAL**, indicates that Moab should not delay the start time of a job if the desired node set is not available but adequate idle resources exist outside of the set. Setting this parameter to *TRUE* basically tells Moab to attempt to use a node set if it is available, but if not, run the job as soon as possible anyway.

**i** Setting **NODESETISOPTIONAL** to *FALSE* will force the job to always run in a complete nodeset regardless of any start delay this imposes.

### Node Set List

Finally, the **NODESETLIST** value of *switchA switchB...* tells Moab to only use node sets based on the listed feature values. This is necessary since sites will often use node features for many purposes and the resulting node sets would be of little use for switch proximity if they were generated based on irrelevant node features indicating things such as processor speed or node architecture.

To add nodes to the **NODESETLIST**, you must configure features on your nodes using the **NODECFG FEATURES** on page 558 attribute.

```
NODECFG[node01] FEATURES=switchA
NODECFG[node02] FEATURES=switchA
NODECFG[node03] FEATURES=switchB
```

*Nodes node01 and node02 contain the switchA feature, and node node03 contains the switchB feature.*

### Node Set Priority

When resources are available in more than one resource set, the **NODESETPRIORITYTYPE** parameter allows control over how the best resource set is selected. Legal values for this parameter are described in the following table:

Priority Type	Description	Details
<b>AFFINITY</b>	Avoid a resource set with negative <a href="#">affinity</a> .	Choosing this type causes Moab to select a node set with no negative affinity nodes (nodes that have a reservation that with negative affinity). If all node sets have negative affinity, then Moab will select the first matching node set.

Priority Type	Description	Details
<b>BESTFIT</b>	Select the smallest resource set possible.	Choosing this type causes Moab, when selecting a node set, to eliminate sets that do not have all the required resources. From the remaining sets, Moab chooses the set with the least amount of resources. This priority type most closely matches the job requirements in order to waste the least amount of resources.  This type minimizes fragmentation of larger resource sets.
<b>MINLOSS</b>	Select the resource set that results in the minimal wasted resources assuming no internal job load balancing is available. (Assumes parallel jobs only run as fast as the slowest allocated node.)	Choosing this type works only when using the following configuration: <b>NODESETATTRIBUTE FEATURE</b> In a <i>SHAREDMEM</i> environment (See <a href="#">Moab-NUMA Integration Guide on page 1255</a> for more information.), Moab will select the node set based on NUMA properties (the smallest feasible node set).
<b>WORSTFIT</b>	Select the largest resource set possible.	This type causes Moab, when choosing a node set, to eliminate sets that do not have all the required resources. From the remaining sets, Moab chooses the set with the greatest amount of resources.  This type minimizes fragmentation of smaller resource sets, but increases fragmentation of larger resource sets.

## Dynamic example

In this example, a site wants to be able to dynamically specify which *VARATTR* values the node set will be based on. To accomplish this, they could use the following configuration in the `moab.cfg` file:

```
NODESETISOPTIONAL FALSE
NODESETPOLICY      FIRSTOF
NODESETATTRIBUTE   VARATTR
```

### Node Set Attribute

The example's [NODESETATTRIBUTE](#) parameter is set to *VARATTR* specifying that the node sets are to be constructed by job *VARATTR* values that are specified dynamically in the `msub` command.

### Node Set Policy

In the preceding example, the [NODESETPOLICY](#) parameter is set to the policy *FIRSTOF* and tells Moab to allocate nodes from the first set that matches specified constraints.

### Node Set Constraint Handling

The parameter, [NODESETISOPTIONAL](#), indicates that Moab should not delay the start time of a job if the desired node set is not available but adequate idle resources exist outside of the set. Setting this

parameter to *FALSE* will force the job to always run in a complete node set regardless of any start delay this imposes.

### *msub example*

With the configuration (above) set in the `moab.cfg`, Moab is configured for dynamic node sets. You can create node sets dynamically by using the `msub -l` command. (For more information, see [Resource Manager Extensions on page 618](#).) Use the following format:

```
msub -l nodeset=FIRSTOF:VARATTR:<var>[=<value>],...
```

For example, if you wanted to create a dynamic node set for the Provo datacenter:

```
msub -l nodeset=FIRSTOF:VARATTR: datacenter=Provo
```

*This command causes Moab to set datacenter=Provo as the node set.*

**i** You can specify more than one VARATTR in the command. For example, if you want to create a dynamic node set for the Provo datacenter and the SaltLake datacenter:

```
msub -l nodeset=FIRSTOF:VARATTR: datacenter=Provo: datacenter=SaltLake
```

If you specify only `datacenter` (without specifying a value, such as `=Provo`), Moab will look up all possible values (values reported on the node for that VARATTR), and then choose one. So if, for example, you have nodes that have VARATTRs `datacenter=Provo`, `datacenter=SaltLake`, and `datacenter=StGeorge`, then specifying `msub -l nodeset=FIRSTOF:VARATTR:datacenter` will cause the job to run in Provo *or* SaltLake *or* StGeorge.

You should also note that Moab also adds the VARATTR (whether you specify it or if Moab chooses it) to the required attribute (REQATTR) of the job. For example, if you specify `datacenter=Provo` as the VARATTR, `datacenter=Provo` will also be added to the job REQATTR. Likewise, if you specify only `datacenter`, and Moab chooses `datacenter=SaltLake`, then `datacenter=SaltLake` will be added to the job REQATTR.

If you do not request a VARATTR in the nodeset of the `msub -l` command, the job will run as if it did not use node sets at all, and nothing will be added to its REQATTR.

**i** If you manually specify a different REQATTR on a job (for example, `datacenter=SaltLake`) from the node set VARATTR (for example, `datacenter=Provo`), the job will never run.

## NODESETPLUS

Moab supports additional NodeSet behavior by specifying the [NODESETPLUS](#) parameter. Possible values when specifying this parameter are *SPANEVENLY* and *DELAY*.

**i** Neither *SPANEVENLY* nor *DELAY* will work with multi-req jobs or preemption.

Value	Description
<b>SPANEVENLY</b>	Moab attempts to fit all jobs within one node set, or it spans any number of node sets evenly. When a job specifies a <a href="#">NODESETDELAY</a> , Moab attempts to contain the job within a single node set; if unable to do so, it spans node sets evenly, unless doing so would delay the job beyond the requested <b>NODESETDELAY</b> .
<b>DELAY</b>	Moab attempts to schedule the job within a nodeset for the configured <a href="#">NODESETDELAY</a> . If Moab cannot find space for the job to start within <b>NODESETDELAY</b> (Moab considers future workload to determine if space will open up in time and might create a future reservation), then Moab schedules the job and ignores the nodeset requirement.

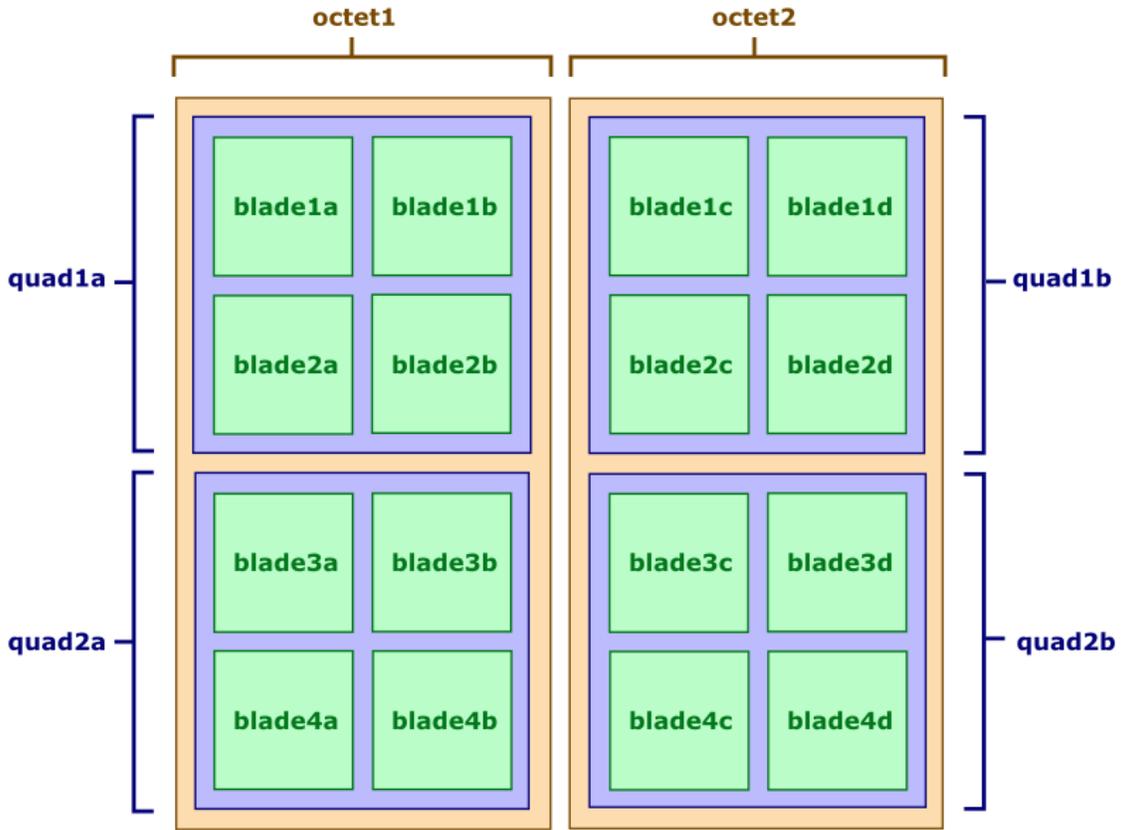
### Nested Node Sets

Moab attempts to fit jobs on node sets in the order they are specified in the **NODESETLIST**. You can create nested node sets by listing your node sets in a specific order. Here is an example of a "smallest to largest" nested node set:

```
NODESETPOLICY ONEOF
NODESETATTRIBUTE FEATURE
NODESETISOPTIONAL FALSE
NODESETLIST blade1a,blade1b,blade2a,blade2b,blade3a,
blade3b,blade4a,blade4b,quad1a,quad1b,quad2a,
quad2b,octet1,octet2,sixteen
```

The accompanying cluster would look like this:

Image 3-8: Octet, quad, and blade node sets on a cluster



In this example, Moab tries to fit the job on the nodes in the blade sets first. If that doesn't work, it moves up to the nodes in the quad sets (a set of four blade sets). If the quads are insufficient, it tries the nodes in the octet sets (a set of four quad node sets).

## Requesting Node Sets for Job Submission

On a per job basis, each user can specify the equivalent of all parameters except **NODESETDELAY**. As mentioned previously, this is accomplished using the [resource manager extensions](#).

## Configuring Node Sets for Classes

Classes can be configured with a default node set. In the configuration file, specify **DEFAULT.NODESET** with the following syntax: `DEFAULT.NODESET=<SETTYPE>:<SETATTR>[:<SETLIST>[,<SETLIST>]...]`. For example, in a heterogeneous cluster with two different types of processors, the following configuration confines jobs assigned to the `amd` class to run on either `ATHLON` or `OPTERON` processors:

```
CLASSCFG [amd] DEFAULT.NODESET=ONEOF:FEATURE:ATHLON,OPTERON
...
```

### Related topics

- [Resource Manager Extensions](#)
- [CLASSCFG](#)
- [Partition Overview](#)

## Evaluating System Performance - Statistics, Profiling, Testing, and Simulation

- [Moab Performance Evaluation Overview on page 521](#)
- [Accounting: Job and System Statistics on page 521](#)
- [Testing New Versions and Configurations on page 523](#)
- [Answering What If? Questions with the Simulator on page 524](#)

### Moab Performance Evaluation Overview

Moab Workload Manager tracks numerous performance statistics for jobs, accounting, users, groups, accounts, classes, QoS, the system, and so forth. These statistics can be accessed through various commands or [Moab Cluster Manager/Monitor](#).

### Accounting: Job and System Statistics

Moab provides extensive accounting facilities that allow resource usage to be tracked by resources (compute nodes), jobs, users, and other objects. The accounting facilities may be used in conjunction with, and correlated with, the accounting records provided by the resource and allocation manager.

Moab maintains both raw persistent data and a large number of processed in memory statistics allowing instant summaries of cycle delivery and system utilization. With this information, Moab can assist in accomplishing any of the following tasks:

- Determining cumulative cluster performance over a fixed time frame.
- Graphing changes in cluster utilization and responsiveness over time.
- Identifying which compute resources are most heavily used.
- Charting resource usage distribution among users, groups, projects, and classes.
- Determining allocated resources, responsiveness, and failure conditions for jobs completed in the past.
- Providing real-time statistics updates to external accounting systems.

This section describes how to accomplish each of these tasks using Moab tools and accounting information.

- [Accounting Overview](#)
- [Real-Time Statistics](#)
- [FairShare Usage Statistics](#)

## Accounting Overview

Moab provides accounting data correlated to most major objects used within the cluster scheduling environment. These records provide job and reservation accounting, resource accounting, and credential-based accounting.

### Job and Reservation Accounting

As each job or reservation completes, Moab creates a complete persistent trace record containing information about who ran, the time frame of all significant events, and what resources were allocated. In addition, actual execution environment, failure reports, requested service levels, and other pieces of key information are also recorded. A complete description of each accounting data field can be found within section [Workload Traces](#).

### Resource Accounting

The load on any given node is available historically allowing identification of not only its usage at any point in time, but the actual jobs which were running on it. [Moab Cluster Manager](#) can show load information (assuming load is configured as a generic metric), but not the individual jobs that were running on a node at some point in the past. For aggregated, historical statistics covering node usage and availability, the `showstats` command may be run with the `-n` flag.

### Credential Accounting

Current and historical usage for users, groups, account, QoSs, and classes are determined in a manner similar to that available for evaluating nodes. For aggregated, historical statistics covering credential usage and availability, the `showstats` command may be run with the corresponding credential flag.

If needed, detailed credential accounting can also be enabled globally or on a credential by credential basis. With detailed credential accounting enabled, real-time information regarding per-credential usage over time can be displayed. To enable detailed per credential accounting, the **ENABLEPROFILING** attribute must be specified for credentials that are to be monitored. For example, to track detailed credentials, the following should be used:

```
USERCFG [DEFAULT]      ENABLEPROFILING=TRUE
QOSCFG [DEFAULT]      ENABLEPROFILING=TRUE
CLASSCFG [DEFAULT]    ENABLEPROFILING=TRUE
GROUPCFG [DEFAULT]    ENABLEPROFILING=TRUE
ACCOUNTCFG [DEFAULT]  ENABLEPROFILING=TRUE
```

Credential level profiling operates by maintaining a number of time-based statistical records for each credential. The parameters [PROFILECOUNT](#) and [PROFILEDURATION](#) control the number and duration of the statistical records.

## Real-Time Statistics

Moab provides real-time statistical information about how the machine is running from a scheduling point of view. The [showstats](#) command is actually a suite of commands providing detailed information on an overall scheduling basis as well as a per user, group, account and node basis. This command gets its information from in memory statistics that are loaded at scheduler start time from the scheduler checkpoint file. (See [Checkpoint/Restart](#) for more information.) This checkpoint file is updated periodically and when the scheduler is shut down allowing statistics to be collected over an extended time frame. At any time, real-time statistics can be reset using the [mschedctl -f](#) command.

In addition to the `showstats` command, the [showstats -f](#) command also obtains its information from the in memory statistics and checkpoint file. This command displays a processor-time based matrix of scheduling performance for a wide variety of metrics. Information such as backfill effectiveness or average job queue time can be determined on a job size/duration basis.

## FairShare Usage Statistics

Regardless of whether fairshare is enabled, detailed credential based fairshare statistics are maintained. Like job traces, these statistics are stored in the directory pointed to by the [STATDIR](#) parameter. Fairshare stats are maintained in a separate statistics file using the format `FS.<EPOCHTIME>` (`FS.982713600`, for example) with one file created per fairshare window. (See the [Fairshare Overview](#) for more information.) These files are also flat text and record credential based usage statistics. Information from these files can be seen via the [mdiag -f](#) command.

### Related topics

- [Simulation Overview](#)
- [Generic Consumable Resources](#)
- [Object Variables](#)
- [Generic Event Counters](#)

## Testing New Versions and Configurations

- [MONITOR Mode](#)
- [INTERACTIVE Mode](#)

## MONITOR Mode

Moab supports a scheduling mode called *MONITOR*. In this mode, the scheduler initializes, contacts the resource manager and other peer services, and conducts scheduling cycles exactly as it would if running in *NORMAL* or production mode. Job are prioritized, reservations created, policies and limits enforced, and administrator and end-user commands enabled. The key difference is that although live resource management information is loaded, *MONITOR* mode disables Moab's ability to start, preempt, cancel, or otherwise modify jobs or resources. Moab continues to attempt to schedule exactly as it would in *NORMAL* mode but its ability to actually impact the system is disabled. Using this mode, a site can quickly verify correct resource manager configuration and scheduler operation. This mode can also be used to validate new policies and constraints. In fact, Moab can be run in *MONITOR* mode on a production system while another scheduler or even another version of Moab is running on the same

system. This unique ability can allow new versions and configurations to be fully tested without any exposure to potential failures and with no cluster downtime.

To run Moab in *MONITOR* mode, simply set the **MODE** attribute of the **SCHEDCFG** parameter to *MONITOR* and start Moab. Normal scheduler commands can be used to evaluate configuration and performance. [Diagnostic commands](#) can be used to look for any potential issues. Further, the Moab log file can be used to determine which jobs Moab attempted to start, and which resources Moab attempted to allocate.

If another instance of Moab is running in production and a site administrator wants to evaluate an alternate configuration or new version, this is easily done but care should be taken to avoid conflicts with the primary scheduler. Potential conflicts include statistics files, logs, checkpoint files, and user interface ports. One of the easiest ways to avoid these conflicts is to create a new test directory with its own log and stats subdirectories. The new `moab.cfg` file can be created from scratch or based on the existing `moab.cfg` file already in use. In either case, make certain that the **PORT** attribute of the **SCHEDCFG** parameter differs from that used by the production scheduler by at least two ports. If testing with the production binary executable, the `MOABHOMEDIR` environment variable should be set to point to the new test directory to prevent Moab from loading the production `moab.cfg` file.

## INTERACTIVE Mode

*INTERACTIVE* mode allows for evaluation of new versions and configurations in a manner different from *MONITOR* mode. Instead of disabling all resource and job control functions, Moab sends the desired change request to the screen and asks for permission to complete it. For example, before starting a job, Moab may print something like the following to the screen:

```
Command: start job 1139.ncsa.edu on node list test013,test017,test018,test021
Accept: (y/n) [default: n]?
```

The administrator must specifically accept each command request after verifying it correctly meets desired site policies. Moab then executes the specified command. This mode is highly useful in validating scheduler behavior and can be used until configuration is appropriately tuned and all parties are comfortable with the scheduler's performance. In most cases, sites will want to set the scheduling mode to *NORMAL* after verifying correct behavior.

### Related topics

- [Testing New Releases and Policies](#)
- [Cluster Simulations](#)
- [Side-by-Side Mode](#)

## Answering *What If?* Questions with the Simulator

Moab Workload Manager can answer hypothetical situations through simulations. (See [Simulations](#).) Once Resource and Workload Traces have been collected, any number of configurations can be tested without disturbing the system.

# General Job Administration

- [Job Holds](#) on page 525
- [Job Priority Management](#) on page 526
- [Suspend/Resume Handling](#) on page 527
- [Checkpoint/Restart Facilities](#) on page 528
- [Job Dependencies](#) on page 528
- [Job Defaults and Per Job Limits](#) on page 530
- [General Job Policies](#) on page 531
- [Using a Local Queue](#) on page 539
- [Job Deadlines](#) on page 542
- [Job Arrays](#) on page 545

## Job Holds

### Holds and Deferred Jobs

Moab supports job holds applied by users ([user holds](#)), administrators ([system holds](#)), and resource managers ([batch holds](#)). There is also a temporary hold known as a [job defer](#).

### User Holds

User holds are very straightforward. Many, if not most, resource managers provide interfaces by which users can place a hold on their own job that tells the scheduler not to run the job while the hold is in place. Users may use this capability because the job's data is not yet ready, or they want to be present when the job runs to monitor results. Such user holds are created by, and under the control of a non-privileged user and may be removed at any time by that user. As would be expected, users can only place holds on their jobs. Jobs with a user hold in place will have a Moab state of `Hold` or `UserHold` depending on the resource manager being used.

### System Holds

The system hold is put in place by a system administrator either manually or by way of an automated tool. As with all holds, the job is not allowed to run so long as this hold is in place. A batch administrator can place and release system holds on any job regardless of job ownership. However, unlike a user hold, normal users cannot release a system hold even on their own jobs. System holds are often used during system maintenance and to prevent particular jobs from running in accordance with current system needs. Jobs with a system hold in place will have a Moab state of `Hold` or `SystemHold` depending on the resource manager being used.

## Batch Holds

Batch holds are placed on a job by the scheduler itself when it determines that a job cannot run. The reasons for this vary but can be displayed by issuing the `checkjob<JOBID>` command. Possible reasons are included in the following list:

- No Resources — The job requests resources of a type or amount that do not exist on the system.
- System Limits — The job is larger or longer than what is allowed by the specified system policies.
- Bank Failure — The allocations bank is experiencing failures.
- No Allocations — The job requests use of an account that is out of allocations and no fallback account has been specified.
- RM Reject — The resource manager refuses to start the job.
- RM Failure — The resource manager is experiencing failures.
- Policy Violation — The job violates certain throttling policies preventing it from running now and in the future.
- No QoS Access — The job does not have access to the QoS level it requests.

Jobs which are placed in a batch hold will show up within Moab in the state `BatchHold`.

## Job Defer

In most cases, a job violating these policies is not placed into a batch hold immediately; rather, it is deferred. The parameter `DEFERTIME` indicates how long it is deferred. At this time, it is allowed back into the idle queue and again considered for scheduling. If it again is unable to run at that time or at any time in the future, it is again deferred for the timeframe specified by `DEFERTIME`. A job is released and deferred up to `DEFERCOUNT` times at which point the scheduler places a batch hold on the job and waits for a system administrator to determine the correct course of action. Deferred jobs have a Moab state of `Deferred`. As with jobs in the `BatchHold` state, the reason the job was deferred can be determined by use of the `checkjob` command.

At any time, a job can be released from any hold or deferred state using the `releasehold` command. The Moab logs should provide detailed information about the cause of any batch hold or job deferral.

**i** Under Moab, the reason a job is deferred or placed in a batch hold is stored in memory but is not checkpointed. Thus this information is available only until Moab is recycled at which point the `checkjob` command no longer displays this reason information.

### Related topics

- `DEFERSTARTCOUNT` - number of job start failures allowed before job is deferred

## Job Priority Management

Job priority management is controlled via both configured and manual intervention mechanisms.

- Priority Configuration - see [Job Prioritization](#)
- Manual Intervention with [setspri](#)

## Suspend/Resume Handling

When supported by the resource manager, Moab can suspend and resume jobs. A user can suspend his/her own jobs, but only an administrator can resume them. By default, a job is suspended for one minute before it can resume. You can modify this default time using the [MINADMINSTIME](#) parameter.

A job must be marked as `suspendable` for Moab to suspend and resume it. To do so, either submit the job with the `suspendable` flag attached to it or configure a credential to pass the flag to its associated jobs. These methods are demonstrated in the examples below:

```
msub -l flags=suspendable
```

```
GROUPCFG[default] JOBFLAGS=SUSPENDABLE
```

Once the job is suspendable, Moab allows you to suspend jobs using the two following methods: (1) manually on the command line and (2) automatically in the `moab.cfg` file.

To manually suspend jobs, use the [mjobctl](#) command as demonstrated in the following example:

```
> mjobctl -s job05
```

*Moab suspends job05, preventing it from running immediately in the job queue.*

If you are an administrator and want to resume a job, use the `mjobctl` command as demonstrated in the following example:

```
> mjobctl -r job05
```

*Moab removes job05 from a suspended state and allows it to run.*

You can also configure the Moab preemption policy to suspend and resume jobs automatically by setting the [PREEMTPOLICY](#) parameter to `SUSPEND`. A sample Moab configuration looks like this:

```
PREEMTPOLICY SUSPEND
...
USERCFG[tom] JOBFLAGS=SUSPENDABLE
```

Moab suspends jobs submitted by user `tom` if necessary to make resources available for jobs with higher priority.



If your resource manager has a native interface, you must configure [JOBSUSPENDURL](#) to suspend and resume jobs.

For more information about suspending and resuming jobs in Moab, see the following sections:

- manual preemption with the [mjobctl](#) command
- [Job preemption](#)

## Checkpoint/Restart Facilities

Checkpointing records the state of a job, allowing for it to restart later without interruption to the job's execution. Checkpointing can be performed manually, as the result of [triggers](#) or [events](#), or in conjunction with various [QoS](#) policies.

Moab's ability to checkpoint is dependent upon both the cluster's [resource manager](#) and operating system. In most cases, two types of checkpoint are enabled, including (1) checkpoint and continue and (2) checkpoint and terminate. While either checkpointing method can be activated using the [mjobctl](#) command, only the checkpoint and terminate type is used by internal scheduling and event managements facilities.

Checkpointing behavior can be configured on a per-resource manager basis using various attributes of the [RMCFG](#) parameter.

### Related topics

- [Job Preemption Overview](#)
- [PREEMTPOLICY](#) Parameter
- Resource Manager [CHECKPOINTSIG](#) Attribute
- Resource Manager [CHECKPOINTTIMEOUT](#) Attribute

## Job Dependencies

- [Basic Job Dependency Support](#)
  - [Job Dependency Syntax](#)

### Basic Job Dependency Support

By default, basic single step job dependencies are supported through completed/failed step evaluation. Basic dependency support does not require special configuration and is activated by default. Dependent jobs are only supported through a resource manager and therefore submission methods depend upon the specific resource manager being used. For the [TORQUE qsub](#) command, the semantics listed in the section below can be used with the `-W x=depend=<STRING>` or `-W depend=<STRING>` flag; for the Moab [msub](#) command, the `-l depend=<STRING>` or `-W x=depend=<STRING>` flag. For other resource managers, consult the resource manager specific documentation.

**i** If you are submitting a job with a dependency in a grid environment via `msub`, you need to use the `-l` option instead of `-w`. You must always use `-W depend=` or `-W x=depend=` for `qsub` dependencies.

## Job Dependency Syntax

Dependency	Format	Description
<b>after</b>	<code>after:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time after specified jobs have started execution.
<b>afterany</b>	<code>afterany:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time after all specified jobs have completed regardless of completion status.
<b>afterok</b>	<code>afterok:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may be start at any time after all specified jobs have successfully completed.
<b>afternotok</b>	<code>afternotok:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time after all specified jobs have completed unsuccessfully.
<b>before</b>	<code>before:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time before specified jobs have started execution.
<b>beforeany</b>	<code>beforeany:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time before all specified jobs have completed regardless of completion status.
<b>beforeok</b>	<code>beforeok:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time before all specified jobs have successfully completed.
<b>beforenotok</b>	<code>beforenotok:&lt;job&gt;</code> <code>[:&lt;job&gt;]...</code>	Job may start at any time before any specified jobs have completed unsuccessfully.
<b>on</b>	<code>on:&lt;count&gt;</code>	Job may start after <i>&lt;count&gt;</i> dependencies on other jobs have been satisfied.
<b>synccount</b>	<code>synccount:&lt;count&gt;</code>	Job is the first in a set of jobs to be executed at the same time. <i>&lt;count&gt;</i> is the number of additional jobs in the set, which can be up to 5. synccount is valid for single-request jobs with TORQUE as the resource manager.
<b>syncwith</b>	<code>syncwith:&lt;job&gt;</code>	Job is an additional member of a set of jobs to be executed at the same time. Moab supports up to 5 jobs. syncwith is valid for single-request jobs with TORQUE as the resource manager.

**i** `<job>={JOBNAME.jobname|jobid}`

When using JobName dependencies, prepend "JOBNAME." to avoid ambiguity.

**i** The `before` dependencies do not work with jobs submitted with `msub`; they work only with `qsub`.

Any of the dependencies containing `before` must be used in conjunction with the `on` dependency. So, if job A must run before job B, job B must be submitted with `depend=on:1`, as well as job A having `depend=before:A`. This means job B cannot run until one dependency of another job on job B has been fulfilled. This prevents job B from running until job A can be successfully submitted.

When you submit a dependency job and the dependency is not met, the job will remain idle in the queue indefinitely. To configure Moab to automatically cancel these failed dependency jobs, set the [CANCELFAILEDDEPENDENCYJOBS](#) on [page 1390](#) scheduler flag.

Related topics

- [Job Deadlines](#)

## Job Defaults and Per Job Limits

### Job Defaults

Job defaults can be specified on a per queue basis. These defaults are specified using the [CLASSCFG](#) parameter. The following table shows the applicable attributes:

Attribute	Format	Example
<b>DEFAULT.FEATURES</b>	comma-delimited list of node features	<pre>CLASSCFG[batch] DEFAULT.FEATURES=fast,io</pre> <p><i>Jobs submitted to class batch will request nodes features <b>fast</b> and <b>io</b>.</i></p>
<b>DEFAULT.WCLIMIT</b>	[[DD:]HH:]MM:]SS	<pre>CLASSCFG[batch] DEFAULT.WCLIMIT=1:00:00</pre> <p><i>Jobs submitted to class batch will request one hour of walltime by default.</i></p>

### Per Job Maximum Limits

Job maximum limits can be specified on a per queue basis. These defaults are specified using the [CLASSCFG](#) parameter. The following table shows the applicable attributes:

Attribute	Format	Example
<b>MAX.WCLIMIT</b>	[[ [DD:]HH:]MM:] SS	<pre>CLASSCFG[batch] MAX.WCLIMIT=1:00:00</pre> <p><i>Jobs submitted to class batch can request no more than one hour of walltime.</i></p>

## Per Job Minimum Limits

Furthermore, minimum job defaults can be specified with the [CLASSCFG](#) parameter. The following table shows the applicable attributes:

Attribute	Format	Example
<b>MIN.PROC</b>	<integer>	<pre>CLASSCFG[batch] MIN.PROC=10</pre> <p><i>Jobs submitted to class batch can request no less than ten processors.</i></p>

Related topics

- [Usage-based Limits](#)

## General Job Policies

- [Multi-Node Support](#)
- [Multi-Req Support](#)
- [Job Size Policy](#)
- [Malleable Job Support](#)
- [Enabling Job User Proxy](#)

There are a number of configurable policies that help control advanced job functions. These policies help determine allowable job sizes and structures.

### Multi-Node Support

You can configure the ability to allocate resources from multiple nodes to a job with the [MAX.NODE](#) limit.

## Multi-Req Support

Jobs can specify multiple types of resources for allocation. For example, a job could request 4 nodes with 256 MB of memory and 8 nodes with feature `fast` present.

Resources specified in a multi-req job are delimited with a plus sign (+).



Neither *SPANEVENLY* nor *DELAY* values of the NODESETPLUS parameter will work with multi-req jobs or preemption.

*Example 3-130:*

```
-l nodes=4:ppn=1+10:ppn=5+2:ppn=2
```

*This example requests 4 nodes with 1 proc each, 10 nodes with 5 procs each, and 2 nodes with 2 procs each. The total number of processors requested is  $(4*1) + (10*5) + (2*2)$ , or 58 processors.*

*Example 3-131:*

```
-l nodes=15+1:ppn=4
```

*The job submitted in this example requests a total of 16 nodes. 15 of these nodes have no specific requirements, but the remaining node must have 4 processors.*

*Example 3-132:*

```
-l nodes=3:fast+1:io
```

*The job requests a total of 4 nodes: 3 nodes with the **fast** feature and 1 node with the **io** feature.*

## Job Size Policy

Moab allows jobs to request resource ranges. Using this range information, the scheduler is able to maximize the amount of resources available to the job while minimizing the amount of time the job is blocked waiting for resources. The [JOBSIZEPOLICY](#) parameter can be used to set this behavior according to local site needs.

**i** Job resource ranges may only be specified when using a local queue as described in the [Using a Local Queue](#) section.

## Malleable Job Support

A job can specify whether it is able to use more processors or less processors and what effect, if any, that has on its wallclock time. For example, a job may run for 10 minutes on 1 processor, 5 minutes on 2 processors and 3 minutes on 3 processors. When a job is submitted with a task request list attached, Moab determines which task request fits best and molds the job based on its specifications. To submit a job with a task request list and allow Moab to mold it based on the current scheduler environment, use the [TRL](#) flag in the Resource Manager Extension.

## Enabling Job User Proxy

By default, user proxying is disabled. To be enabled, it must be authorized using the **PROXYLIST** attribute of the [USERCFG](#) parameter. This parameter can be specified either as a comma-delimited list of users or as the keyword *validate*. If the keyword *validate* is specified, the [RMCFG](#) attribute **JOBVALIDATEURL** should be set and used to confirm that the job's owner can proxy to the job's execution user. An example script performing this check for ssh-based systems is provided in the `tools` directory (See [Job Validate Tool Overview](#)).

For some resource managers (RM), proxying must also be enabled at the RM level. The following example shows how ssh-based proxying can be accomplished in a Moab+TORQUE with SSH environment.

**i** To validate proxy users, Moab must be running as root.

## Example 3-133: SSH Proxy Settings

```
USERCFG[DEFAULT] PROXYLIST=validate
RMCFG[base] TYPE=<resource manager>
JOBVALIDATEURL=exec://$HOME/tools/job.validate.sshproxy.pl
```

```
> qmgr -c 's s allow proxy user=true'
> su - testuser
> qsub -I -u testuser2
qsub: waiting for job 533.igt.org to start
qsub: job 533.igt.org ready
testuser2@igt:~$
```

*In this example, the validate tool, 'job.validate.sshproxy.pl', can verify proxying is allowed by becoming the submit user and determining if the submit user can achieve passwordless access to the specified execution user. However, site-specific tools can use any method to determine proxy access including a flat file look-up, database lookup, querying of an information service such as NIS or LDAP, or other local or remote tests. For example, if proxy validation is required but end-user accounts are not available on the management node running Moab, the job validate service could perform the validation test on a representative remote host such as a login host.*

 This feature supports qsub only.

The job validate tool is highly flexible allowing any combination of job attributes to be evaluated and tested using either local or remote validation tests. The validate tool allows not only pass/fail responses but also allows the job to be modified, or rejected in a custom manner depending on the site or the nature of the failure.

## Related topics

- [Usage Limits](#)

## Using a Local Queue

Moab allows jobs to be submitted directly to the scheduler. With a local queue, Moab is able to directly manage the job or translate it for resubmission to a standard resource manager queue. There are multiple advantages to using a local queue:

- Jobs may be translated from one resource manager job submission language to another (such as submitting a PBS job and running it on an LSF cluster).
- Jobs may be migrated from one local resource manager to another.
- Jobs may be migrated to remote systems using Moab peer-to-peer functionality.
- Jobs may be dynamically modified and optimized by Moab to improve response time and system utilization.
- Jobs may be dynamically modified to account for system hardware failures or other issues.
- Jobs may be dynamically modified to conform to site policies and constraints.
- Grid jobs are supported.

## Local Queue Configuration

A local queue is configured just like a standard resource manager queue. It may have defaults, limits, resource mapping, and credential access constraints. The following table describes the most common settings:

Default queue	
<b>Format</b>	<code>RMCFG[internal] DEFAULTCLASS=&lt;CLASSID&gt;</code>
<b>Description</b>	<p>The job class/queue assigned to the job if one is not explicitly requested by the submitter.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> All jobs submitted directly to Moab are initially received by the pseudo-resource manager <i>internal</i>. Therefore, default queue configuration may only be applied to it.</p> </div>
<b>Example</b>	<code>RMCFG[internal] DEFAULTCLASS=batch</code>

Class default resource requirements	
<b>Format</b>	<code>CLASSCFG[&lt;CLASSID&gt;] DEFAULT.FEATURES=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;]</code> <code>DEFAULT.MEM=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] DEFAULT.NODE=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;]</code> <code>DEFAULT.NODESET=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] DEFAULT.PROC=&lt;X&gt; CLASSCFG</code> <code>[&lt;CLASSID&gt;] DEFAULT.WCLIMIT=&lt;X&gt;</code>
<b>Description</b>	The settings assigned to the job if not explicitly set by the submitter. Default values are available for node features, per task memory, node count, nodeset configuration, processor count, and wallclock limit.
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <code>CLASSCFG[batch] DEFAULT.WCLIMIT=4 DEFAULT.FEATURES=matlab</code> </div> <p>or</p> <div style="border: 1px dashed #ccc; padding: 5px;"> <code>CLASSCFG[batch] DEFAULT.WCLIMIT=4</code>  <code>CLASSCFG[batch] DEFAULT.FEATURES=matlab</code> </div>

Class maximum resource limits	
<b>Format</b>	<code>CLASSCFG[&lt;CLASSID&gt;] MAX.FEATURES=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] MAX.NODE=&lt;X&gt;</code> <code>CLASSCFG[&lt;CLASSID&gt;] MAX.PROC=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] MAX.WCLIMIT=&lt;X&gt;</code>

### Class maximum resource limits

<b>Description</b>	The maximum node features, node count, processor count, and wallclock limit allowed for a job submitted to the class/queue. If these limits are not satisfied, the job is not accepted and the submit request fails. <b>MAX.FEATURES</b> indicates that only the listed features may be requested by a job.
<b>Example</b>	<pre>CLASSCFG[smalljob] MAX.PROC=4 MAX.FEATURES=slow,matlab</pre> <p>or</p> <pre>CLASSCFG[smalljob] MAX.PROC=4 CLASSCFG[smalljob] MAX.FEATURES=slow,matlab</pre>

### Class minimum resource limits

<b>Format</b>	<pre>CLASSCFG[&lt;CLASSID&gt;] MIN.FEATURES=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] MIN.NODE=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] MIN.PROC=&lt;X&gt; CLASSCFG[&lt;CLASSID&gt;] MIN.WCLIMIT=&lt;X&gt;</pre>
<b>Description</b>	The minimum node features, node count, processor count, and wallclock limit allowed for a job submitted to the class/queue. If these limits are not satisfied, the job is not accepted and the submit request fails. <b>MIN.FEATURES</b> indicates that only the listed features may be requested by a job.
<b>Example</b>	<pre>CLASSCFG[bigjob] MIN.PROC=4 MIN.WCLIMIT=1:00:00</pre> <p>or</p> <pre>CLASSCFG[bigjob] MIN.PROC=4 CLASSCFG[bigjob] MIN.WCLIMIT=1:00:00</pre>

### Class access

<b>Format</b>	<pre>CLASSCFG[&lt;CLASSID&gt;] REQUIREUSERLIST=&lt;USERID&gt; [, &lt;USERID&gt;] ...</pre>
<b>Description</b>	The list of users who may submit jobs to the queue.
<b>Example</b>	<pre>CLASSCFG[math] REQUIREUSERLIST=john,steve</pre>

Available resources	
<b>Format</b>	CLASSCFG [<CLASSID>] HOSTLIST=<HOSTID>[, <HOSTID>] . . .
<b>Description</b>	The list of nodes that jobs in the queue may use.
<b>Example</b>	<code>CLASSCFG[special] HOSTLIST=node001,node003,node13</code>

Class mapping between multiple sites is described in the section on Moab grid facilities.

If a job is submitted directly to the resource manager used by the local queue, the class default resource requirements are not applied. Also, if the job violates a local queue limitation, the job is accepted by the resource manager, but placed in the Blocked state.

## Job Deadlines

- [Deadline Overview](#)
- [Setting Job Deadlines via QoS on page 542](#)
  - [Setting Job Deadlines at Job Submission on page 543](#)
  - [Submitting a Job to a QoS with a Preconfigured Deadline on page 543](#)
- [Job Termination Date](#)
- [Conflict Policies](#)

### Deadline Overview

Job deadlines may be specified on a per job and per credential basis and are also supported using both absolute and QoS based specifications. A job requesting a deadline is first evaluated to determine if the deadline is acceptable. If so, Moab adds it to the list of deadline jobs and allocates resources to guarantee that all accepted deadline jobs are able to complete on or before their requested deadline. Once the scheduler confirms that all deadlines can be satisfied, it then optimizes resource allocation (in priority order) attempting to execute all jobs at the earliest possible time.

### Setting Job Deadlines via QoS

Two types of job deadlines exist in Moab. The priority-based deadline linearly increases a job's priority as its deadline approaches (See [Deadline \(DEADLINE\) Subcomponent on page 387](#) for more information). The QoS method allows you to set a job completion time on job submission if, and only if, it requests and is allowed to access a QoS with the [DEADLINE QFLAG](#) set. This method is more powerful than the priority method, because Moab will attempt to make a reservation for the job as soon as the job enters the queue in order to meet the deadline, essentially bumping it to the front of the queue.

When a job is submitted to a QoS with the **DEADLINE** flag set, the job's `-l deadline` attribute is honored. If such QoS access is not available, or if resources do not exist at job submission time to allow the deadline to be satisfied, the job's deadline request is ignored.

Two methods exist for setting deadlines with a QoS:

- Submitting a job to a deadline-enabled QoS and specifying a deadline using `msub -l`.
- Submitting a job to a deadline-enabled QoS with a **QTTARGET** specified.

## Setting Job Deadlines at Job Submission

This method of setting a job deadline allows you to specify a job deadline as you submit the job. You can set the deadline as either an exact date and time or as an amount of time after job submission (i.e. three hours after submission).

### To specify a deadline on job submission

1. In `moab.cfg`, create a QoS with the **DEADLINE** flag enabled.

```
...
QOSCFG[special] QFLAGS=DEADLINE
```

*Jobs requesting the QoS *special* may submit jobs with a deadline that Moab will honor.*

2. Submit a job to the QoS and set a deadline. This can be either absolute or relative.

- a. For an absolute deadline, use the format `hh:mm:ss_mm/dd/yy`. The following configuration sets a deadline for a job to finish by 8 a.m. on March 15th, 2013.

```
msub -l qos=special deadline=08:00:00_03/15/13 job.sh
```

*The job must finish running by 8 A.M. on March 15, 2013.*

- b. For a relative deadline, or the completion deadline of the job relative to its submission time, use the time format `[ [DD:]HH:]MM:]SS`.

```
msub -l qos=special deadline=5:00:00 job.sh
```

*The job's deadline is 5 hours after its submission.*

## Submitting a Job to a QoS with a Preconfigured Deadline

You may also set a relative job deadline by limiting the job's queue time. This method allows you to preconfigure the deadline rather than giving the power to specify a deadline to the user submitting the job. For jobs requesting these QoSes, Moab identifies and sets job deadlines to satisfy the corresponding response time targets.

### To submit a job to a QoS with a preconfigured deadline

1. In `moab.cfg`, create a QoS with both the **DEADLINE QFLAG** and a response time target (**QTTARGET**). The **QTTARGET** is the maximum amount of time that Moab should allow the job to be idle in the

queue.

```
...
QOSCFG[special2] QFLAGS=DEADLINE QTTARGET=1:00:00
```

*Given this configuration, a job requesting QoS special2 must spend a maximum of one hour in the queue.*

2. Submit a job requesting the special2 quality of service.

```
msub -l qos=special2 walltime=2:00:00 job.sh
```

*This two-hour job has a completion time deadline set to three hours after its submission (one hour of target queue time and two hours of run time).*

## Job Termination Date

In addition to job completion targets, jobs may also be submitted with a [TERMTIME](#) attribute. The scheduler attempts to complete the job prior to the termination date, but if it is unsuccessful, it will terminate (cancel) the job once the termination date is reached.

## Conflict Policies

The specific policy can be configured using the [DEADLINEPOLICY](#) parameter. Moab does not have a default policy for this parameter.

Policy	Description
<b>CANCEL</b>	The job is canceled and the user is notified that the deadline could not be satisfied.
<b>HOLD</b>	The job has a batch hold placed on it indefinitely. The administrator can then decide what action to take.
<b>RETRY</b>	The job continually retries each iteration to meet its deadline; note that when used with <code>QTTARGET</code> the job's deadline continues to slide with relative time.
<b>IGNORE</b>	The job has its request ignored and is scheduled as normal.

**i** Deadline scheduling may not function properly with per partition scheduling enabled. Check that **PARALLELATIONPOLICY** is disabled to ensure **DEADLINEPOLICY** will work correctly.

### Related topics

- [QoS Facilities](#)
- Job Submission [Eligible Start Time](#) constraints

## Job Arrays

- [Job Array Overview](#)
- [Enabling Job Arrays](#)
- [Sub-job Definitions](#)
- [Using Environment Variables to Specify Array Index Values](#)
  - [Control](#)
  - [Reporting](#)
- [Job Array Cancellation Policies](#)
- [Examples](#)
  - [Submitting Job Arrays](#)

### Job Array Overview

You can submit an array of jobs to Moab via the [msub](#) command. Array jobs are an easy way to submit many sub-jobs that perform the same work using the same script, but operate on different sets of data. Sub-jobs are the jobs created by an array job and are identified by the array job ID and an index; for example, if `235[1]` is an identifier, the number 235 is a job array ID, and 1 is the sub-job.

Sub-jobs of an array are executed in sub-job index order.



Moab job arrays are different from TORQUE job arrays.

### Enabling Job Arrays

To enable job arrays, include the [ENABLEJOBARRAYS](#) parameter in the Moab configuration file (`moab.cfg`).

### Sub-job Definitions

Like a normal job, an array job submits a job script, but it additionally has a start index (`sidx`) and an end index (`eidx`); array jobs also have increment (`incr`) values, which Moab uses to create sub-jobs, all executing the same script. The model for sub-job creation follows the formula of end index minus start index plus increment divided by the increment value:  $(eidx - sidx + incr) / incr$ .

To illustrate, suppose an array job has a start index of 1, an end index of 100, and an increment of 1. This is an array job that creates  $(100 - 1 + 1) / 1 = 100$  sub-jobs with indexes of 1, 2, 3, ..., 100. An increment of 2 produces  $(100 - 1 + 2) / 2 = 50$  sub-jobs with indexes of 1, 3, 5, ..., 99. An increment of 2 with a start index of 2 produces  $(100 - 2 + 2) / 2 = 50$  sub-jobs with indexes of 2, 4, 6, ..., 100. Again, sub-jobs are jobs in their own right that have a slightly different job naming convention `jobID[subJobIndex]` (e.g. `mycluster.45[37]` or `45[37]`).

## Using Environment Variables to Specify Array Index Values

The script can use an environment variable to obtain the array index value to form data file and/or directory names unique to an array job's particular sub-job. The following two environment variables are supplied so job scripts can recognize what index in the array they are in; use the [msub](#) command with the [-V](#) option to pass the environment parameters to the resource manager, or include the parameters in a job script; for example: `#PBS -V MOAB_JOBARRAYRANGE`.

Environment Parameter	Description
<b>MOAB_JOBARRAYINDEX</b>	<p>Used to create dataset file names, directory names, and so forth, when splitting up a single problem into multiple jobs.</p> <p>For example, a user may split up a problem into 20 separate jobs, each with its own input and output data files whose names contain the numbers 1-20.</p> <p>To illustrate, assume a user submits the 20 sub-jobs using two msub commands; one to submit the ten even-numbered jobs and one to submit the ten odd-numbered jobs.</p> <pre>msub -t job1. [1-20:2] msub -t job2. [2-20:2]</pre> <p>The MOAB_JOBARRAYINDEX environment variable value would populate each of the two job arrays' ten sub-jobs as 1, 3, 5, 7, 9, 11, 13, 15, 17 and 19 for the first array job's ten sub-jobs, and 2, 4, 6, 8, 10, 12, 14, 16, 18, and 20 for the second array job's ten sub-jobs.</p>
<b>MOAB_JOBARRAYRANGE</b>	The count of jobs in the array.

### Control

Users can control individual sub-jobs in the same manner as normal jobs. In addition, an array job represents its group of sub-jobs and any user or administrator commands performed on an array job apply to its sub-jobs; for example, the command [canceljob <arrayJobId>](#) cancels all sub-jobs that belong to the array job. For more information about job control, see the documentation for the [mjobctl](#) command.

### Reporting

In the first example below, the parts unique to array subjobs are in red.

```

$ checkjob -v Moab.1[1]
job Moab.1[1]

AName: Moab
State: Running
Creds: user:user1 group:usergroup1
WallTime: 00:00:17 of 8:20:00
SubmitTime: Thu Nov 4 11:50:03
(Time Queued Total: 00:00:00 Eligible: INFINITY)
StartTime: Thu Nov 4 11:50:03
Total Requested Tasks: 1
Req[0] TaskCount: 1 Partition: base
Average Utilized Procs: 0.96
NodeCount: 1
Allocated Nodes:
[node010:1]

Job Group:      Moab.1
Parent Array ID: Moab.1
Array Index:    1
Array Range:    10
SystemID:      Moab
SystemJID:     Moab.1[1]
Task Distribution: node010
IWD:           /home/user1
UMask:         0000
Executable:    /opt/moab/spool/moab.job.3CvNj1
StartCount:    1
Partition List: base
SrcRM:         internal DstRM: base DstRMJID: Moab.1[1]
Flags:         ARRAYJOB,GLOBALQUEUE
StartPriority: 1
PE:           1.00
Reservation 'Moab.1[1]' (-00:00:19 -> 8:19:41 Duration: 8:20:00)

```

If the array range is not provided, the output displays all the jobs in the array.

```
$ checkjob -v Moab.1
job Moab.1

AName: Moab
Job Array Info:
  Name: Moab.1
  1 : Moab.1[1] : Running
  2 : Moab.1[2] : Running
  3 : Moab.1[3] : Running
  4 : Moab.1[4] : Running
  5 : Moab.1[5] : Running
  6 : Moab.1[6] : Running
  7 : Moab.1[7] : Running
  8 : Moab.1[8] : Running
  9 : Moab.1[9] : Running
 10 : Moab.1[10] : Running
 11 : Moab.1[11] : Running
 12 : Moab.1[12] : Running
 13 : Moab.1[13] : Running
 14 : Moab.1[14] : Running
 15 : Moab.1[15] : Running
 16 : Moab.1[16] : Running
 17 : Moab.1[17] : Running
 18 : Moab.1[18] : Running
 19 : Moab.1[19] : Running
 20 : Moab.1[20] : Running
Totals:
  Active: 20
  Idle: 0
  Complete: 0
```

You can also use [showq](#). This displays the array master job with a count of how many sub-jobs are in each queue.

```

$ showq

active jobs-----
JOBID            USERNAME        STATE  PROCS   REMAINING        STARTTIME
Moab.1(5)        aesplin         Running  5      00:52:41  Thu Jun 23 17:05:56
Moab.2(1)        aesplin         Running  1      00:53:41  Thu Jun 23 17:06:56

6 active jobs                6 of 6 processors in use by local jobs (100.00%)
1 of 1 nodes active          (100.00%)

eligible jobs-----
JOBID            USERNAME        STATE  PROCS   WCLIMIT        QUEUE TIME
Moab.2(4)        aesplin         Idle    4      1:00:00  Thu Jun 23 17:06:56

4 eligible jobs

blocked jobs-----
JOBID            USERNAME        STATE  PROCS   WCLIMIT        QUEUE TIME
Moab.2(1)        aesplin         Blocked  1      1:00:00  Thu Jun 23 17:06:56

1 blocked job

Total jobs: 11
    
```

*Moab.1 has five sub-jobs running. Moab.2 has one sub-job running, four waiting to run, and one that is currently blocked.*

## Job Array Cancellation Policies

Job arrays can be canceled based on the success or failure of the first sub-job, the first success or failure of any sub-job, or if any sub-job exits with a specified exit code. The job array cancellation policies are:

Cancel Policy	Description	Exclusivity
<b>CancelOnFirstFailure</b>	Cancels the job array if the first sub-job ( <b>JOBARRAYINDEX = 1</b> ) fails. <pre>&gt; msub -t myarray[1-1000]%50 -1 ...,flags=CancelOnFirstFailure</pre>	Mutually exclusive
<b>CancelOnFirstSuccess</b>	Cancels the job array if the first sub-job ( <b>JOBARRAYINDEX = 1</b> ) succeeds. <pre>&gt; msub -t myarray[1-1000]%50 -1 ...,flags=CancelOnFirstSuccess</pre>	
<b>CancelOnAnyFailure</b>	Cancels the job array if any sub-job fails. <pre>&gt; msub -t myarray[1-1000]%50 -1 ...,flags=CancelOnAnyFailure</pre>	

Cancel Policy	Description	Exclusivity
<b>CancelOnAnySuccess</b>	<p>Cancels the job array if any sub-job succeeds.</p> <pre>&gt; msub -t myarray[1-1000]%50 -l ...,flags=CancelOnAnySuccess</pre>	
<b>CancelOnExitCode</b>	<p>Cancels the job array if any sub-job returns the specified exit code.</p> <pre>&gt; msub -t myarray[1-1000%50] -l ...,flags=CancelOnExitCode:&lt;error code list&gt;</pre> <p>The syntax for the error code list are ranges specified with a dash and individual codes delimited by a plus (+) sign, such as: 1-4+9+15 Exit codes 1-387 are accepted.</p>	

Up to two cancellation policies can be specified for an array and the two policies must be delimited by a colon (:). The two "first sub-job" policies are mutually exclusive, as are the three "any sub-job" policies. You can use either "first sub-job" policy with one of the "any sub-job" policies, as shown in this example:

```
> msub -t myarray[1-1000]%50 -l ...,flags=CancelOnFirstFailure:CancelOnExitCode:3-7+11
```

## Examples

Operations can be performed on individual jobs, a selection of jobs in a job array, or on the entire array.

### Submitting Job Arrays

The syntax for submitting job arrays is: `msub -t [<jobname>]<indexlist>[%<limit>] arrayscript.sh`

The `<jobname>` and `<limit>` are optional. The jobname does not override the `jobID` Moab assigns to the array. When submitting an array with a jobname, Moab returns the `jobID`, which is the scheduler name followed by a unique ID.

For example, if the scheduler name in `moab.cfg` is *Moab* (`SCHEDCFG[Moab]`), submitting an array with a jobname responds like this:

```
> msub -t myarray[1-10] job.sh
Moab.6
```

To specify that only a certain number of sub-jobs in the array can run at a time, use the percent sign (%) delimiter. In this example, only five sub-jobs in the array can run at a time:

```
> msub -t myarray[1-1000]%5
```

To submit a specific set of array sub-jobs, use the comma delimiter in the array index list:

```
> msub -t myarray[1,2,3,4]
> msub -t myarray[1-5,7,10]
```

You can use the `checkjob` command on either the `jobID` or the `jobname` you specified.

```
> msub -t myarray[1-2] job.sh

Moab.10

$ checkjob -v myarray
  job Moab.10

AName: myarray
Job Array Info:
  Name: Moab.10
  1 : Moab.10[1] : Running
  2 : Moab.10[2] : Running

Sub-jobs:          2
Active:            2 ( 100.0% )
Eligible:          0 ( 0.0% )
Blocked:           0 ( 0.0% )
Completed:         0 ( 0.0% )

State: Idle
Creds: user:tuser1 group:tgroup1
WallTime: 00:00:00 of 99:23:59:59
SubmitTime: Thu Jun 2 16:37:17
          (Time Queued Total: 00:00:33 Eligible: 00:00:00)

Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: ALL
```

To submit a job with a step size, use a colon in the array range and specify how many jobs to step. In the example below, a step size of 2 is requested. The sub-jobs will be numbered according to the step size inside the index limit. The array master job name will be the same as explained above.

```

$ msub -t myarray[2-10:2] job.sh

job Moab.15

$ checkjob -v myarray #or you could use 'checkjob -v Moab.15'
job Moab.15

AName: myarray
Job Array Info:
  Name: Moab.15
  2 : Moab.15[2] : Running
  4 : Moab.15[4] : Running
  6 : Moab.15[6] : Running
  8 : Moab.15[8] : Running
 10 : Moab.15[10] : Running

Sub-jobs:          5
Active:            5 ( 100.0% )
Eligible:          0 (  0.0% )
Blocked:           0 (  0.0% )
Completed:         0 (  0.0% )

State: Idle
Creds: user:tuser1 group:tgroup1
WallTime: 00:00:00 of 99:23:59:59
SubmitTime: Thu Jun  2 16:37:17
  (Time Queued Total: 00:00:33 Eligible: 00:00:00)

Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: ALL

```

### Related topics

- [Moab Workload Manager for Grids](#)
- [Job Dependencies](#)

## General Node Administration

- [Node Location on page 553](#)
- [Node Attributes on page 556](#)
- [Node Specific Policies on page 566](#)
- [Managing Shared Cluster Resources \(Floating Resources\) on page 567](#)
- [Managing Node State on page 571](#)
- [Managing Consumable Generic Resources on page 573](#)
- [Enabling Generic Metrics on page 575](#)
- [Enabling Generic Events on page 578](#)

## Overview

Moab has a very flexible and generalized definition of a [node](#). This flexible definition, together with the fact that Moab must inter-operate with many resource managers of varying capacities, requires that Moab must possess a complete set of mechanisms for managing nodes that in some cases may be redundant with resource manager facilities.

### Resource Manager Specified 'Opaque' Attributes

Many resource managers support the concept of opaque node attributes, allowing a site to assign arbitrary strings to a node. These strings are opaque in the sense that the resource manager passes them along to the scheduler without assigning any meaning to them. Nodes possessing these opaque attributes can then be requested by various jobs. Using certain Moab parameters, sites can assign a meaning within Moab to these opaque node attributes and extract specific node information. For example, setting the parameter [FEATUREPROCSPEEDHEADER](#) `xps` causes a node with the opaque string `xps950` to be assigned a processor speed of 950 MHz within Moab.

### Scheduler Specified Default Node Attributes

Some default node attributes can be assigned on a rack or partition basis. In addition, many node attributes can be specified globally by configuring the *DEFAULT* node template using the [NODECFG](#) parameter (i.e., `NODECFG[DEFAULT] PROCSPEED=3200`). Unless explicitly specified otherwise, nodes inherit node attributes from the associated rack or partition or from the default node template. See the [Partition Overview](#) for more information.

### Scheduler Specified Node Attributes

The **NODECFG** parameter also allows direct per-node specification of virtually all node attributes supported via other mechanisms and also provides a number of additional attributes not found elsewhere. For example, a site administrator may want to specify something like the following:

```
NODECFG[node031] MAXJOB=2 PROCSPEED=600 PARTITION=small
```



These approaches may be mixed and matched according to the site's local needs. Precedence for the approaches generally follows the order listed earlier in cases where conflicting node configuration information is specified through one or more mechanisms.

## Node Location

Nodes can be assigned three types of location information based on partitions, racks, and queues.

- [Partitions](#)
- [Racks](#)
- [Queues](#)
  - [TORQUE/OpenPBS Queue to Node Mapping](#)

- [Node Selection/Specification](#)

## Partitions

The first form of location assignment, the partition, allows nodes to be grouped according to physical resource constraints or policy needs. By default, jobs are not allowed to span more than one partition so partition boundaries are often valuable if an underlying network topology make certain resource allocations undesirable. Additionally, per-partition policies can be specified to grant control over how scheduling is handled on a partition by partition basis. See the [Partition Overview](#) for more information.

## Racks

Rack-based location information is orthogonal to the partition based configuration and is mainly an organizational construct. In general rack based location usage, a node is assigned both a rack and a slot number. This approach has descended from the IBM SP2 organizational approach in which a rack can contain any number of slots but typically contains between 1 and 99. Using the rack and slot number combo, individual compute nodes can be grouped and displayed in a more ordered manner in certain Moab commands (i.e., [showstate](#)). Currently, rack information can only be specified directly by the system via the SDR interface on SP2/Loadleveler systems. In all other systems, this information must be specified using an information service or specified manually using the [RACK](#), [SLOT](#), and [SIZE](#) attributes of the [NODECFG](#) parameter.

**i** Sites may arbitrarily assign nodes to racks and rack slots without impacting scheduling behavior. Neither rack numbers nor rack slot numbers need to be contiguous; their use is simply for convenience purposes in displaying and analyzing compute resources.

*Example 3-134:*

```
NODECFG[node024] RACK=1 SLOT=1
NODECFG[node025] RACK=1 SLOT=2
NODECFG[node026] RACK=2 SLOT=1 PARTITION=special
...
```

When specifying node and rack information, slot values must be in the range of 1 to 99, and racks must be in the range of 1 to 399.

## Queues

Some resource managers allow queues (or classes) to be defined and then associated with a subset of available compute resources. With systems such as Loadleveler or PBSPro these queue to node mappings are automatically detected. On resource managers that do not provide this service, Moab provides alternative mechanisms for enabling this feature.

### TORQUE/OpenPBS Queue to Node Mapping

Under [TORQUE](#), queue to node mapping can be accomplished by using the [qmgr](#) command to set the queue [acl\\_hosts](#) parameter to the mapping hostlist desired. Further, the [acl\\_host\\_enable](#) parameter should be set to `False`.

**i** Setting `acl_hosts` and then setting `acl_host_enable` to `True` constrains the list of hosts from which jobs may be submitted to the queue.

The following example highlights this process and maps the queue debug to the nodes `host14` through .

```
> qmgr
Max open servers: 4
Qmgr: set queue debug acl_hosts = "host14,host15,host16,host17"
Qmgr: set queue debug acl_host_enable = false
Qmgr: quit
```

**i** All queues that do not have `acl_hosts` specified are global; that is, they show up on every node. To constrain these queues to a subset of nodes, each queue requires its own `acl_hosts` parameter setting.

## Node Selection

When selecting or specifying nodes either via command line tools or via configuration file based lists, Moab offers three types of node expressions that can be based on node lists, exact lists, node ranges, or regular expressions.

### Node Lists

Node lists can be specified as one or more comma or whitespace delimited node IDs. Specified node IDs can be based on either short or fully qualified hostnames. Each element will be interpreted as a regular expression.

```
SRCFG[basic]  HOSTLIST=cl37.icluster,ax45,ax46
...
```

### Exact Lists

When Moab receives a list of nodes it will, by default, interpret each element as a regular expression. To disable this and have each element interpreted as a string node name, the `l:` can be used as in the following example:

```
> setres l:n00,n01,n02
```

### Node Range

Node lists can be specified as one or more comma or whitespace delimited node ranges. Each node range can be based using either `<STARTINDEX>-<ENDINDEX>` or `<HEADER>[<STARTINDEX>-<ENDINDEX>]` format. To explicitly request a range, the node expression must be preceded with the string `r:` as in the following example:

```
> setres r:37-472,513,516-855
```

When you specify a `<HEADER>` for the range, note that it must only contain alphabetical characters. As always, the range must be numeric.

```
CLASSCFG[long] HOSTLIST=r:anc-b[37-472]
```

**i** Only one expression is allowed with node ranges.

**i** By default, Moab attempts to extract a node's node index assuming this information is built into the node's naming convention. If needed, this information can be explicitly specified in the Moab configuration file using `NODECFG`'s **NODEINDEX** attribute, or it can be extracted from alternately formatted node IDs by specifying the `NODEIDFORMAT` parameter.

## Node Regular Expression

Node lists may also be specified as one or more comma or whitespace delimited regular expressions. Each node regular expression must be specified in a format acceptable by the standard C regular expression libraries that allow support for wildcard and other special characters such as the following:

- \* (asterisk)
- . (period)
- [ ] (left and right bracket)
- ^ (caret)
- \$ (dollar)

Node lists are by default interpreted as a regular expression but can also be explicitly requested with the string `X:` as in the following examples:

```
# select nodes c130 thru c155
SRCFG[basic] HOSTLIST=x:c1[34],c15[0-5]
...
```

```
# select nodes c130 thru c155
SRCFG[basic] HOSTLIST=c1[34],c15[0-5]
...
```

**i** To control node selection search ordering, set the **OBJECTELIST** parameter to one of the following options: `exact`, `range`, `regex`, `rangere`, or `rerange`.

## Node Attributes

- [Configurable Node Attributes on page 556](#)
- [Node Features/Node Properties on page 565](#)

## Configurable Node Attributes

Nodes can possess a large number of attributes describing their configuration which are specified using the `NODECFG` parameter. The majority of these attributes such as operating system or configured

network interfaces can only be specified by the direct resource manager interface. However, the number and detail of node attributes varies widely from resource manager to resource manager. Sites often have interest in making scheduling decisions based on scheduling attributes not directly supplied by the resource manager. Configurable node attributes are listed in the following table; click an attribute for more detailed information:

- |                                             |                                                |                                           |
|---------------------------------------------|------------------------------------------------|-------------------------------------------|
| <a href="#">ACCESS</a> on page 557          | <a href="#">NETWORK</a> on page 559            | <a href="#">PROVRM</a> on page 562        |
| <a href="#">ARCH</a> on page 557            | <a href="#">NODEINDEX</a> on page 559          | <a href="#">RACK</a> on page 562          |
| <a href="#">COMMENT</a> on page 557         | <a href="#">OS</a> on page 560                 | <a href="#">RADISK</a> on page 562        |
| <a href="#">ENABLEPROFILING</a> on page 557 | <a href="#">OSLIST</a> on page 560             | <a href="#">RCDISK</a> on page 563        |
| <a href="#">FEATURES</a> on page 558        | <a href="#">OVERCOMMIT</a> on page 560         | <a href="#">RCMEM</a> on page 563         |
| <a href="#">FLAGS</a> on page 558           | <a href="#">PARTITION</a> on page 560          | <a href="#">RCPROC</a> on page 563        |
| <a href="#">GRES</a> on page 559            | <a href="#">POWERPOLICY</a> on page 560        | <a href="#">RCSWAP</a> on page 564        |
| <a href="#">MAXIOIN</a> on page 559         | <a href="#">PREEMPTMAXCPULOAD</a> on page 560  | <a href="#">SIZE</a> on page 564          |
| <a href="#">MAXJOB</a> on page 559          | <a href="#">PREEMPTMINMEMAVAIL</a> on page 561 | <a href="#">SLOT</a> on page 564          |
| <a href="#">MAXJOBPERUSER</a> on page 559   | <a href="#">PREEMPTPOLICY</a> on page 561      | <a href="#">SPEED</a> on page 564         |
| <a href="#">MAXPE</a> on page 559           | <a href="#">PRIORITY</a> on page 561           | <a href="#">TRIGGER</a> on page 564       |
| <a href="#">MAXPEPERJOB</a> on page 559     | <a href="#">PRIORITYF</a> on page 562          | <a href="#">VARIABLE</a> on page 564      |
| <a href="#">MAXPROC</a> on page 559         | <a href="#">PROCSPEED</a> on page 562          | <a href="#">VMOCTHRESHOLD</a> on page 565 |

Attribute	Description
<b>ACCESS</b>	Specifies the node access policy that can be one of <i>SHARED</i> , <i>SHAREDONLY</i> , <i>SINGLEJOB</i> , <i>SINGLETASK</i> , or <i>SINGLEUSER</i> . See <a href="#">Node Access Policies</a> for more details. <pre>{ NODECFG[node013] ACCESS=singlejob }</pre>
<b>ARCH</b>	Specifies the node's processor architecture. <pre>{ NODECFG[node013] ARCH=opteron }</pre>
<b>COMMENT</b>	Allows an organization to annotate a node via the configuration file to indicate special information regarding this node to both users and administrators. The <b>COMMENT</b> value may be specified as a quote delimited string as shown in the example that follows. Comment information is visible using <a href="#">checknode</a> , <a href="#">mdiag</a> , <a href="#">Moab Cluster Manager</a> , and <a href="#">Moab Access Portal</a> . <pre>{ NODECFG[node013] COMMENT="Login Node" }</pre>
<b>ENABLEPROFILING</b>	Allows an organization to track node state over time. This information is available using <a href="#">showstats -n</a> . <pre>{ NODECFG[DEFAULT] ENABLEPROFILING=TRUE }</pre>

Attribute	Description
<p><b>FEATURES</b></p>	<p>Not all resource managers allow specification of opaque <a href="#">node features</a> (also known as node properties). For these systems, the <a href="#">NODECFG</a> parameter can be used to directly assign a list of node features to individual nodes. To append node features, use <code>FEATURES=&lt;X&gt;</code>; to overwrite or remove a node's features, you must update them in your Moab configuration file or resource manager.</p> <pre data-bbox="540 495 1409 541">NODECFG[node013] FEATURES=gpfs,fastio</pre> <p data-bbox="586 558 1360 636"><i>Node node013 now has features <b>gpfs</b> and <b>fastio</b> in addition to any other features configured in this file or the resource manager.</i></p> <div data-bbox="540 663 1409 751"> <p><b>i</b> The total number of supported node features is limited as described in the <a href="#">Adjusting Default Limits</a> section.</p> </div> <div data-bbox="540 768 1409 951"> <p><b>i</b> If supported by the resource manager, the resource manager specific manner of requesting node features/properties within a job may be used. (Within <b>TORQUE</b>, use <code>qsub -l nodes=&lt;NODECOUNT&gt;:&lt;NODEFEATURE&gt;</code>.) However, if either not supported within the resource manager or if support is limited, the Moab <a href="#">feature</a> resource manager extension may be used.</p> </div>
<p><b>FLAGS</b></p>	<p>Specifies various flags that should be set on the given node. Node flags must be set using the <a href="#">mschedctl -m config</a> command. Do not set node flags in the <code>moab.cfg</code> file. Flags set in <code>moab.cfg</code> may conflict with settings controlled automatically by resource managers, Moab Web Services.</p> <ul style="list-style-type: none"> <li>• <b>globalvars</b> - The node has variables that may be used by triggers.</li> <li>• <b>novmmigrations</b> - Excludes this hypervisor from VM auto-migrations. This means that VMs cannot automatically migrate to or from this hypervisor while this flag is set.</li> </ul> <pre data-bbox="618 1287 1409 1333">NODECFG[node1] FLAGS=NoVMMigrations</pre> <p data-bbox="667 1350 1360 1581"><i>To allow VMs to resume migrating, remove this flag using <code>mschedctl -m config 'NODECFG[node1] FLAGS=NoVMMigrations'</code> or use a resource manager to unset the flag. Because both Moab and the RM report the <b>novmmigration</b> flag and the RM's setting always overrides the Moab setting, you cannot remove the flag via the Moab command when the RM is reporting it.</i></p>

Attribute	Description
<p><b>GRES</b></p>	<p>Many resource managers do not allow specification of consumable generic node resources. For these systems, the <a href="#">NODECFG</a> parameter can be used to directly assign a list of consumable generic attributes to individual nodes or to the special pseudo-node <a href="#">global</a>, which provides shared cluster (floating) consumable resources. To set/overwrite a node's generic resources, use <code>GRES=&lt;NAME&gt;[:&lt;COUNT&gt;]</code>. (See <a href="#">Managing Consumable Generic Resources</a>.)</p> <pre data-bbox="565 527 1433 573">NODECFG [node013] GRES=quickcalc:20</pre>
<p><b>MAXIOIN</b></p>	<p>Maximum input allowed on node before it is marked busy.</p>
<p><b>MAXJOB</b></p>	<p>See <a href="#">Node Policies</a> for details.</p>
<p><b>MAXJOBPERUSER</b></p>	<p>See <a href="#">Node Policies</a> for details.</p>
<p><b>MAXPE</b></p>	<p>See <a href="#">Node Policies</a> for details.</p>
<p><b>MAXPEPERJOB</b></p>	<p>Maximum allowed Processor Equivalent per job on this node. A job will not be allowed to run on this node if its PE exceeds this number.</p> <pre data-bbox="565 1010 1433 1083">NODECFG [node024] MAXPEPERJOB=10000 ...</pre>
<p><b>MAXPROC</b></p>	<p>Maximum dedicated processors allowed on this node. No jobs are scheduled on this node when this number is reached. See <a href="#">Node Policies</a> for more information.</p> <pre data-bbox="565 1209 1433 1283">NODECFG [node024] MAXPROC=8 ...</pre>
<p><b>NETWORK</b></p>	<p>The ability to specify which networks are available to a given node is limited to only a few resource managers. Using the <b>NETWORK</b> attribute, administrators can establish this node to network connection directly through the scheduler. The <a href="#">NODECFG</a> parameter allows this list to be specified in a comma-delimited list.</p> <pre data-bbox="565 1472 1433 1545">NODECFG [node024] NETWORK=GigE ...</pre>
<p><b>NODEINDEX</b></p>	<p>The node's index. See <a href="#">Node Location</a> for details.</p>

Attribute	Description
<p><b>OS</b></p>	<p>This attribute specifies the node's operating system.</p> <pre data-bbox="537 369 1406 415">NODECFG[node013] OS=suse10</pre> <div data-bbox="537 443 1406 562" style="border: 1px solid #005596; padding: 5px; margin-top: 10px;"> <p><b>i</b> Because the TORQUE operating system overwrites the Moab operating system, change the operating system with <a href="#">opsys on page 2794</a> instead of <b>OS</b> if you are using TORQUE.</p> </div>
<p><b>OSLIST</b></p>	<p>This attribute specifies the list of operating systems the node can run.</p> <pre data-bbox="537 663 1406 709">NODECFG[compute002] OSLIST=linux,windows</pre>
<p><b>OVERCOMMIT</b></p>	<p>Specifies the high-water limit for over-allocation of processors or memory on a hypervisor. This setting is used to protect hypervisors from having too many VMs placed on them, regardless of the utilization level of those VMs. Possible attributes include DISK, MEM, PROC, and SWAP. Usage is <code>&lt;attr&gt;:&lt;integer&gt;</code>.</p> <pre data-bbox="537 905 1406 951">NODECFG[node012] OVERCOMMIT=PROC:2,MEM:4</pre>
<p><b>PARTITION</b></p>	<p>See <a href="#">Node Location</a> for details.</p>
<p><b>POWERPOLICY</b></p>	<p>The <b>POWERPOLICY</b> can be set to <i>OnDemand</i> or <i>STATIC</i>. It defaults to <i>STATIC</i> if not set. If set to <i>STATIC</i>, Moab will never automatically change the power status of a node. If set to <i>OnDemand</i>, Moab will turn the machine off and on based on workload and global settings. See Green Computing for further details.</p>
<p><b>PREEMPTMAXCPULOAD</b></p>	<p>If the node CPU load exceeds the specified value, any batch jobs running on the node are preempted using the preemption policy specified with the node's <a href="#">PREEMTPOLICY</a> attribute. If this attribute is not specified, the global default policy specified with <a href="#">PREEMTPOLICY</a> parameter is used. See <a href="#">Sharing Server Resources</a> for further details.</p> <pre data-bbox="537 1430 1406 1514">NODECFG[node024] PRIORITY=-150 COMMENT="NFS Server Node" NODECFG[node024] PREEMTPOLICY=CANCEL PREEMPTMAXCPULOAD=1.2 ...</pre>

Attribute	Description
<p><b>PREEMPTMINMEMAVAIL</b></p>	<p>If the available node memory drops below the specified value, any batch jobs running on the node are preempted using the preemption policy specified with the node's <a href="#">PREEMTPOLICY</a> attribute. If this attribute is not specified, the global default policy specified with <a href="#">PREEMTPOLICY</a> parameter is used. See <a href="#">Sharing Server Resources</a> for further details.</p> <pre data-bbox="565 499 1433 590"> NODECFG[node024] PRIORITY=-150 COMMENT="NFS Server Node" NODECFG[node024] PREEMTPOLICY=CANCEL PREEMPTMINMEMAVAIL=256 ...</pre>
<p><b>PREEMTPOLICY</b></p>	<p>If any node preemption policies are triggered (such as <a href="#">PREEMPTMAXCPULOAD</a> or <a href="#">PREEMPTMINMEMAVAIL</a>) any batch jobs running on the node are preempted using this preemption policy if specified. If not specified, the global default preemption policy specified with <a href="#">PREEMTPOLICY</a> parameter is used. See <a href="#">Sharing Server Resources</a> for further details.</p> <pre data-bbox="565 821 1433 911"> NODECFG[node024] PRIORITY=-150 COMMENT="NFS Server Node" NODECFG[node024] PREEMTPOLICY=CANCEL PREEMPTMAXCPULOAD=1.2 ...</pre>
<p><b>PRIORITY</b></p>	<p>The <b>PRIORITY</b> attribute specifies the fixed node priority relative to other nodes. It is only used if <a href="#">NODEALLOCATIONPOLICY</a> is set to <i>PRIORITY</i>. The default node priority is 0. A default cluster-wide node priority may be set by configuring the <b>PRIORITY</b> attribute of the <i>DEFAULT</i> node. See <a href="#">Priority Node Allocation</a> for more details.</p> <pre data-bbox="565 1104 1433 1194"> NODEALLOCATIONPOLICY PRIORITY NODECFG[node024] PRIORITY=120 ...</pre>

Attribute	Description
<p><b>PRIORITYF</b></p>	<p>The <b>PRIORITYF</b> attribute specifies the function to use when calculating a node's allocation priority specific to a particular job. It is only used if <a href="#">NODEALLOCATIONPOLICY</a> is set to <i>PRIORITY</i>. The default node priority function sets a node's priority exactly equal to the configured node priority. The priority function allows a site to indicate that various environmental considerations such as node load, reservation affinity, and ownership be taken into account as well using the following format:</p> <pre>&lt;COEFFICIENT&gt; * &lt;ATTRIBUTE&gt; [ + &lt;COEFFICIENT&gt; * &lt;ATTRIBUTE&gt; ] ...</pre> <p>&lt;ATTRIBUTE&gt; is an attribute from the table found in the <a href="#">Priority Node Allocation</a> section.</p> <p>A default cluster-wide node priority function may be set by configuring the <b>PRIORITYF</b> attribute of the <i>DEFAULT</i> node. See <a href="#">Priority Node Allocation</a> for more details.</p> <pre>NODEALLOCATIONPOLICY  PRIORITY NODECFG[node024]  PRIORITYF='APROC + .01 * AMEM - 10 * JOBCOUNT' ...</pre>
<p><b>PROCSPEED</b></p>	<p>Knowing a node's processor speed can help the scheduler improve intra-job efficiencies by allocating nodes of similar speeds together. This helps reduce losses due to poor internal job load balancing. Moab's <a href="#">Node Set</a> scheduling policies allow a site to control processor speed based allocation behavior.</p> <p>Processor speed information is specified in MHz and can be indicated directly using <a href="#">NODECFG</a> or through use of the <a href="#">FEATUREPROCSPEEDHEADER</a> parameter.</p>
<p><b>PROVRM</b></p>	<p>Provisioning resource managers can be specified on a per node basis. This allows flexibility in mixed environments. If the node does not have a provisioning resource manager, the default provisioning resource manager will be used. The default is always the first one listed in <code>moab.cfg</code>.</p> <pre>RMCFG[prov]  TYPE=NATIVE  RESOURCETYPE=PROV RMCFG[prov]  PROVDURATION=10:00 RMCFG[prov]  NODEMODIFYURL=exec://\$HOME/tools/os.switch.pl ... NODECFG[node024]  PROVRM=prov</pre>
<p><b>RACK</b></p>	<p>The rack associated with the node's physical location. Valid values range from 1 to 400. See <a href="#">Node Location</a> for details.</p>
<p><b>RADISK</b></p>	<p>Jobs can request a certain amount of disk space through the <a href="#">RM Extension String's DDISK</a> parameter. When done this way, Moab can track the amount of disk space available for other jobs. To set the total amount of disk space available the <b>RADISK</b> parameter is used.</p>

Attribute	Description
<p><b>RCDISK</b></p>	<p>Jobs can request a certain amount of disk space (in MB) through the <a href="#">RM Extension String's DDISK</a> parameter. When done this way, Moab can track the amount of disk space available for other jobs. The <b>RCDISK</b> attribute constrains the amount of disk reported by a resource manager while the <b>RADISK</b> attribute specifies the amount of disk available to jobs. If the resource manager does not report available disk, the <b>RADISK</b> attribute should be used.</p>
<p><b>RCMEM</b></p>	<p>Jobs can request a certain amount of real memory (RAM) in MB through the <a href="#">RM Extension String's DMEM</a> parameter. When done this way, Moab can track the amount of memory available for other jobs. The <b>RCMEM</b> attribute constrains the amount of RAM reported by a resource manager while the <b>RAMEM</b> attribute specifies the amount of RAM available to jobs. If the resource manager does not report available disk, the <b>RAMEM</b> attribute should be used.</p> <p>Please note that memory reported by the resource manager will override the configured value unless a trailing caret (^) is used.</p> <div data-bbox="565 835 1433 905" style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>NODECFG [node024] RCMEM=2048 ...</pre> </div> <div data-bbox="613 911 1385 999" style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <p><i>If the resource manager does not report any memory, then Moab will assign node0242048 MB of memory.</i></p> </div> <div data-bbox="565 1024 1433 1094" style="border: 1px dashed gray; padding: 5px; margin-bottom: 10px;"> <pre>NODECFG [node024] RCMEM=2048^ ...</pre> </div> <div data-bbox="613 1100 1385 1188" style="border: 1px dashed gray; padding: 5px;"> <p><i>Moab will assign 2048 MB of memory to node024 regardless of what the resource manager reports.</i></p> </div>
<p><b>RCPROC</b></p>	<p>The <b>RCPROC</b> specifies the number of processors available on a compute node.</p> <div data-bbox="565 1287 1433 1356" style="border: 1px dashed gray; padding: 5px;"> <pre>NODECFG [node024] RCPROC=8 ...</pre> </div>

Attribute	Description
<p><b>RCSWAP</b></p>	<p>Jobs can request a certain amount of swap space in MB.</p> <div data-bbox="537 359 1409 512" style="border: 1px solid #0070C0; padding: 5px; margin-bottom: 10px;"> <p><b>i</b> <b>RCSWAP</b> works similarly to <b>RCMEM</b>. Setting <b>RCSWAP</b> on a node will set the swap but can be overridden by swap reported by the resource manager. If the trailing caret (^) is used, Moab will ignore the swap reported by the resource manager and use the configured amount.</p> </div> <div data-bbox="537 533 1409 604" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <pre>NODECFG[node024] RCSWAP=2048 ...</pre> </div> <div data-bbox="586 611 1360 699" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <p><i>If the resource manager does not report any memory, Moab will assign node024 <b>2048</b> MB of swap.</i></p> </div> <div data-bbox="537 720 1409 791" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <pre>NODECFG[node024] RCSWAP=2048^ ...</pre> </div> <div data-bbox="586 798 1360 886" style="border: 1px dashed #ccc; padding: 5px;"> <p><i>Moab will assign <b>2048</b> MB of swap to node024 regardless of what the resource manager reports.</i></p> </div>
<p><b>SIZE</b></p>	<p>The number of slots or size units consumed by the node. This value is used in graphically representing the cluster using <a href="#">showstate</a> or <a href="#">Moab Cluster Manager</a>. See <a href="#">Node Location</a> for details. For display purposes, legal size values include <b>1, 2, 3, 4, 6, 8, 12, and 16</b>.</p> <div data-bbox="537 1079 1409 1150" style="border: 1px dashed #ccc; padding: 5px;"> <pre>NODECFG[node024] SIZE=2 ...</pre> </div>
<p><b>SLOT</b></p>	<p>The first slot in the rack associated with the node's physical location. Valid values range from 1 to <b>MMAK_RACKSIZE</b> (default=<b>64</b>). See <a href="#">Node Location</a> for details.</p>
<p><b>SPEED</b></p>	<p>Because today's processors have multiple cores and adjustable clock frequency, this feature has no meaning and will be deprecated.</p> <div data-bbox="537 1379 1409 1440" style="border: 1px solid #0070C0; padding: 5px;"> <p><b>i</b> The <b>SPEED</b> specification must be in the range of 0.01 to 100.0.</p> </div>
<p><b>TRIGGER</b></p>	<p>See <a href="#">About object triggers on page 745</a> for information.</p>
<p><b>VARIABLE</b></p>	<p>Variables associated with the given node, which can be used in job scheduling. See <a href="#">-l PREF</a>.</p> <div data-bbox="537 1646 1409 1717" style="border: 1px dashed #ccc; padding: 5px;"> <pre>NODECFG[node024] VARIABLE=var1 ...</pre> </div>

Attribute	Description
<b>VMOCTHRESHOLD</b>	Specifies the high-water threshold for utilization of resources on a server (i.e. processor and memory). This setting is used to protect hypervisors from becoming too highly utilized and thus negatively impacting the performance of VMs running on the hypervisor. Possible attributes include <code>PROC</code> and <code>MEM</code> .  <code>NODECFG [node024] VMOCTHRESHOLD=PROC=2, MEM=2</code>

## Node Features/Node Properties

A node feature (or node property) is an opaque string label that is associated with a compute node. Each compute node may have any number of node features assigned to it, and jobs may request allocation of nodes that have specific features assigned. Node features are labels and their association with a compute node is not conditional, meaning they cannot be consumed or exhausted.

Node features may be assigned by the resource manager, and this information may be imported by Moab or node features may be specified within Moab directly. Moab supports hyphens and underscores in node feature names.

As a convenience feature, certain node attributes can be specified via node features using the parameters listed in the following table:

PARAMETER	DESCRIPTION
<b>FEATURENODETYPEHEADER</b>	Set <a href="#">Node Type</a>
<b>FEATUREPARTITIONHEADER</b>	Set <a href="#">Partition</a>
<b>FEATUREPROCSPEEDHEADER</b>	Set <a href="#">Processor Speed</a>
<b>FEATURERACKHEADER</b>	Set <a href="#">Rack</a>
<b>FEATURESLOTHEADER</b>	Set <a href="#">Slot</a>

Example 3-135:

```
FEATUREPARTITIONHEADER par
FEATUREPROCSPEEDHEADER cpu
```

### Related topics

- Job [Preferences](#)
- Configuring [Specifying node features \(node properties\)](#) on page 2573 in [TORQUE](#)
- Configuring Node Features in Moab with [NODECFG](#)
- Specifying Job Feature Requirements

- Viewing Feature Availability Breakdown with [mdiag -t](#)
- Differences between Node Features and [Managing Consumable Generic Resources](#)

## Node Specific Policies

Node policies within Moab allow specification of not only how the node's load should be managed, but who can use the node, and how the node and jobs should respond to various events. These policies allow a site administrator to specify on a node by node basis what the node will and will not support. Node policies may be applied to specific nodes or applied system-wide using the specification `NODECFG [DEFAULT] . . . .`

### Node Usage/Throttling Policies

#### MAXJOB

This policy constrains the number of total independent jobs a given node may run simultaneously. It can only be specified via the [NODECFG](#) parameter.

**i** On Cray XT systems, use the NID (node id) instead of the node name. For more information, see [Configuring the moab.cfg file](#).

#### MAXJOBPERUSER

Constrains the number of total independent jobs a given node may run simultaneously associated with any single user. It can only be specified via the [NODECFG](#) parameter.

#### MAXJOBPERGROUP

Constrains the number of total independent jobs a given node may run simultaneously associated with any single group. It can only be specified via the [NODECFG](#) parameter.

#### MAXLOAD

**MAXLOAD** constrains the CPU load the node will support as opposed to the number of jobs. This maximum load policy can also be applied system wide using the parameter [NODEMAXLOAD](#).

#### MAXPE

This policy constrains the number of total dedicated processor-equivalents a given node may support simultaneously. It can only be specified via the **NODECFG** parameter.

#### MAXPROC

This policy constrains the number of total dedicated processors a given node may support simultaneously. It can only be specified via the **NODECFG** parameter.

## MAXPROCUSER

This policy constrains the number of total processors a given node may have dedicated to any single user. It can only be specified via the **NODECFG** parameter.

## MAXPROCERGROUP

This policy constrains the number of total processors a given node may have dedicated to any single group. It can only be specified via the **NODECFG** parameter.

**i** Node throttling policies are used strictly as constraints. If a node is defined as having a single processor or the **NODEACCESSPOLICY** is set to **SINGLETASK**, and a **MAXPROC** policy of 4 is specified, Moab will not run more than one task per node. A node's configured processors must be specified so that multiple jobs may run and then the **MAXJOB** policy will be effective. The number of configured processors per node is specified on a resource manager specific basis. PBS, for example, allows this to be adjusted by setting the number of virtual processors with the `np` parameter for each node in the PBS `nodes` file.

Example 3-136:

```
NODECFG[node024] MAXJOB=4 MAXJOBPERUSER=2
NODECFG[node025] MAXJOB=2
NODECFG[node026] MAXJOBPERUSER=1
NODECFG[DEFAULT] MAXLOAD=2.5
...
```

## Node Access Policies

While most sites require only a single cluster wide node access policy (commonly set using **NODEACCESSPOLICY**), it is possible to specify this policy on a node by node basis using the **ACCESS** attributes of the **NODECFG** parameter. This attribute may be set to any of the valid node access policy values listed in the [Node Access Policies](#) section.

Example 3-137:

To set a global policy of **SINGLETASK** on all nodes except nodes 13 and 14, use the following:

```
# by default, enforce dedicated node access on all nodes
NODEACCESSPOLICY SINGLETASK
# allow nodes 13 and 14 to be shared
NODECFG[node13] ACCESS=SHARED
NODECFG[node14] ACCESS=SHARED
```

Related topics

- [mnodectl](#)

# Managing Shared Cluster Resources (Floating Resources)

This section describes how to configure, request, and reserve cluster file system space and bandwidth, [software licenses](#), and generic cluster resources.

## Shared Cluster Resource Overview

Shared cluster resources such as file systems, networks, and licenses can be managed through creating a pseudo-node. You can configure a pseudo-node via the [NODECFG](#) parameter much as a normal node would be but additional information is required to allow the scheduler to contact and synchronize state with the resource.

In the following example, a license manager is added as a cluster resource by defining the *GLOBAL* pseudo-node and specifying how the scheduler should query and modify its state.

```
NODECFG[GLOBAL] RMLIST=NATIVE
NODECFG[GLOBAL] QUERYCMD=/usr/local/bin/flquery.sh
NODECFG[GLOBAL] MODIFYCMD=/usr/local/bin/flmodify.sh
```

In some cases, pseudo-node resources may be very comparable to node-locked [generic resources](#) however there are a few fundamental differences which determine when one method of describing resources should be used over the other. The following table contrasts the two resource types.

Attribute	Pseudo-Node	Generic Resource
<b>Node-Locked</b>	No - Resources can be encapsulated as an independent node.	Yes - Must be associated with an existing compute node.
<b>Requires exclusive batch system control over resource</b>	No - Resources (such as file systems and licenses) may be consumed both inside and outside of batch system workload.	Yes - Resources must only be consumed by batch workload. Use outside of batch control results in loss of resource synchronization.
<b>Allows scheduler level allocation of resources</b>	Yes - If required, the scheduler can take external administrative action to allocate the resource to the job.	No - The scheduler can only maintain logical allocation information and cannot take any external action to allocate resources to the job.

## Configuring Generic Consumable Floating Resources

Consumable floating resources are configured in the same way as node-locked [generic](#) resources with the exception of using the *GLOBAL* node instead of a particular node.

```
NODECFG[GLOBAL] GRES=tape:4,matlab:2
...
```

*In this setup, four resources of type [tape](#) and 2 of type [matlab](#) are floating and available across all nodes.*

## Requesting Consumable Floating Resources

Floating resources are requested on a per task basis using native resource manager job submission methods or using the [GRES](#) resource manager extensions.

## Configuring Cluster File Systems

Moab allows both the file space and bandwidth attributes of a cluster file system to be tracked, reserved, and scheduled. With this capability, a job or reservation may request a particular quantity of file space and a required amount of I/O bandwidth to this file system. While file system resources are managed as a cluster generic resource, they are specified using the **FS** attribute of the **NODECFG** parameter as in the following example:

```
NODECFG[GLOBAL] FS=PV1:10000@100,PV2:5000@100
...
```

*In this example, PV1 defines a 10 GB file system with a maximum throughput of 100 MB/s while PV2 defines a 5 GB file system also possessing a maximum throughput of 100 MB/s.*

A job may request cluster file system resources using the `fs` resource manager extension. For a TORQUE based system, the following could be used:

```
>qsub -l nodes=1,walltime=1:00:00 -W x=fs:10@50
```

## Configuring Cluster Licenses

Jobs may request and reserve software licenses using native methods or using the **GRES** resource manager extension. If the cluster license manager does not support a query interface, license availability may be specified within Moab using the **GRES** attribute of the **NODECFG** parameter.

*Example 3-138: Configure Moab to support four floating **quickcalc** and two floating **matlab** licenses.*

```
NODECFG[GLOBAL] GRES=quickcalc:4,matlab:2
...
```

*Example 3-139: Submit a **TORQUE** job requesting a node-locked or floating **quickcalc** license.*

```
> qsub -l nodes=1,software=quickcalc,walltime=72000 testjob.cmd
```

## Configuring Generic Resources as Features

Moab can be configured to treat generic resources as features in order to provide more control over server access. For instance, if a node is configured with a certain **GRES** and that **GRES** is turned off, jobs requesting the node will not run. To turn a GRES into a feature, set the **FEATUREGRES** attribute of **GRES\_CFG** to **TRUE** in the `moab.cfg` file.

```
GRES_CFG[gres1] FEATUREGRES=TRUE
```

*Moab now treats gres1 as a scheduler-wide feature rather than a normal generic resource.*

Note that jobs are submitted normally using the same GRES syntax.

**i** If you are running a grid, verify that `FEATUREGRES=TRUE` is set on all members of the grid.

**i** You can safely upgrade an existing cluster to use the feature while jobs are running. If you are in a grid, upgrade all clusters at the same time.

Two methods exist for managing GRES features: via Moab commands and via the resource manager. Using Moab commands means that feature changes are not checkpointed; they do not remain in place when Moab restarts. Using the resource manager causes changes to be reported by the RM, so any changes made before a Moab restart are still present after it.

These methods are mutually exclusive. Use one or the other, but do not mix methods.

## Managing Feature GRES via Moab Commands

In the following example, *gres1* and *gres2* are configured in the `moab.cfg` file. *gres1* is not currently functioning correctly, so it is set to 0, turning the feature off. Values above 0 and non-specified values turn the feature on.

```
NODECFG[GLOBAL] GRES=gres1:0
NODECFG[GLOBAL] GRES=gres2:10000
GRESCFG[gres1] FEATUREGRES=TRUE
GRESCFG[gres2] FEATUREGRES=TRUE
```

*Moab now treats gres1 and gres2 as features.*

To verify that this is set up correctly, run `mdiag -S -v`. It returns the following:

```
> mdiag -S -v
...
Scheduler FeatureGres: gres1:off,gres2:on
```

Once Moab has started, use `mschedctl -m` to modify whether the feature is turned on or off.

```
mschedctl -m sched featuregres:gres1=on
INFO: FeatureGRes 'gres1' turned on
```

You can verify that the feature turned on or off by once again running `mdiag -S -v`.

**i** If Moab restarts, it will not checkpoint the state of these changed feature general resources. Instead, it will read the `moab.cfg` file to determine whether the feature GRES is on or off.

With feature GRES configured, jobs are submitted normally, requesting GRES type *gres1* and *gres2*. Moab ignores GRES counts and reads the feature simply as on or off.

```
> msub -l nodes=1,walltime=600,gres=gres1

1012
> checkjob 1012
job 1012

AName: STDIN
State: Running
.....
StartTime: Tue Jul 3 15:33:28
Feature GRes: gres1
Total Requested Tasks: 1
```

If you request a feature that is currently turned off, the state is not reported as Running, but as Idle. A message like the following returns:

```
BLOCK MSG: requested feature gres 'gres2' is off
```

## Managing Feature GRES via the Resource Manager

You can automate the process of having a feature GRES turn on and off by setting up an external tool and configuring Moab to query the tool the same way that Moab queries a license manager. For example:

```
RMCFG[myRM] CLUSTERQUERYURL=file:/// $HOME/tools/myRM.dat TYPE=NATIVE
RESOURCE TYPE=LICENSE

GRES CFG[gres1] FEATUREGRES=TRUE
GRES CFG[gres2] FEATUREGRES=TRUE
```

*LICENSE means that the RM does not contain any compute resources and that Moab should not attempt to use it to manage any jobs (start, cancel, submit, etc.).*

The `myRM.dat` file should contain something like the following:

```
GLOBAL state=Idle cres=gres1:0,gres2:10
```

External tools can easily update the file based on filesystem availability. Switching any of the feature GRES to 0 turns it off and switching it to a positive value turns it on. If you use this external mechanism, you do not need to use `mschedctl -m` to turn a feature GRES on or off. You also do not need to worry about whether Moab has checkpointed the information or not, since the information is provided by the RM and not by any external commands.

Related topics

- [Managing Resources Directly with the Native Interface](#)

## Managing Node State

There are multiple models in which Moab can operate allowing it to either honor the node state set by an external service or locally determine and set the node state. This section covers the following:

- identifying meanings of particular node states
- specifying node states within locally developed services and resource managers
- adjusting node state within Moab based on load, policies, and events

## Node State Definitions

State	Definition
<b>Down</b>	Node is either not reporting status, is reporting status but failures are detected, or is reporting status but has been marked down by an administrator.
<b>Idle</b>	Node is reporting status, currently is not executing any workload, and is ready to accept additional workload.

State	Definition
<b>Busy</b>	Node is reporting status, currently is executing workload, and cannot accept additional workload due to load.
<b>Running</b>	Node is reporting status, currently is executing workload, and can accept additional workload.
<b>Drained</b>	Node is reporting status, currently is not executing workload, and cannot accept additional workload due to administrative action.
<b>Draining</b>	Node is reporting status, currently is executing workload, and cannot accept additional workload due to administrative action.

## Specifying Node States within Native Resource Managers

Native resource managers can report node state implicitly and explicitly, using **NODESTATE**, **LOAD**, and other attributes. See [Managing Resources Directly with the Native Interface](#) for more information.

## Moab Based Node State Adjustment

Node state can be adjusted based on reported processor, memory, or other load factors. It can also be adjusted based on reports of one or more resource managers in a multi-resource manager configuration. Also, both generic events and generic metrics can be used to adjust node state.

- TORQUE [health scripts](#) (allow compute nodes to detect and report site specific failures).

## Adjusting Scheduling Behavior Based on Reported Node State

Based on reported node state, Moab can support various policies to make better use of available resources. For more information, see the [Green computing overview on page 723](#).

### Down State

- [JOBACTIONONNODEFAILURE](#) parameter (cancel/requeue jobs if allocated nodes fail).
- [Triggers](#) (take specified action if failure is detected).

### Related topics

- [Managing Resources Directly with the Native Interface](#)
- [License Management](#)
- [Adjusting Node Availability](#)
- [NODEMAXLOAD](#) parameter
- [Green computing overview](#)

## Managing Consumable Generic Resources

- Configuring Node-Locked Consumable Generic Resources
  - Requesting Consumable Generic Resources
- Managing Generic Resource Race Conditions

Each time a job is allocated to a compute node, it consumes one or more types of resources. Standard resources such as CPU, memory, disk, network adapter bandwidth, and swap are automatically tracked and consumed by Moab. However, in many cases, additional resources may be provided by nodes and consumed by jobs that must be tracked. The purpose of this tracking may include accounting, billing, or the prevention of resource over-subscription. Generic consumable resources may be used to manage software licenses, I/O usage, bandwidth, application connections, or any other aspect of the larger compute environment; they may be associated with compute nodes, networks, storage systems, or other real or virtual resources.

These additional resources can be managed within Moab by defining one or more generic resources. The first step in defining a generic resource involves naming the resource. Generic resource availability can then be associated with various compute nodes and generic resource usage requirements can be associated with jobs.

### Differences Between Node Features and Consumable Resources

A [node feature](#) (or node property) is an opaque string label that is associated with a compute node. Each compute node may have any number of node features assigned to it and jobs may request allocation of nodes that have specific features assigned. Node features are labels and their association with a compute node is not conditional, meaning they cannot be consumed or exhausted.

### Configuring Node-locked Consumable Generic Resources

Consumable generic resources are supported within Moab using either direct configuration or resource manager auto-detect (as when using TORQUE and [accelerator hardware](#)). For direct configuration, node-locked consumable generic resources (or generic resources) are specified using the **NODECFG** parameter's **GRES** attribute. This attribute is specified using the format `<ATTR>:<COUNT>` as in the following example:

```
NODECFG[titan001] GRES=tape:4
NODECFG[login32] GRES=matlab:2,prime:4
NODECFG[login33] GRES=matlab:2
...
```

 By default, Moab supports up to 128 independent generic resource types.

### Requesting Consumable Generic Resources

Generic resources can be requested on a per task or per job basis using the [GRES resource manager extension](#). If the generic resource is located on a compute node, requests are by default interpreted as a per task request. If the generic resource is located on a shared, cluster-level resource (such as a network or storage system), then the request defaults to a per job interpretation.

**1** Generic resources are specified per task, not per node. When you submit a job, each processor becomes a task. For example, a job asking for `nodes=3:ppn=4,gres=test:5` asks for 60 gres of type test ((3\*4 processors)\*5).

If using [TORQUE](#), the [GRES](#) or [software](#) resource can be requested as in the following examples:

#### Example 3-140: Per Task Requests

```
NODECFG[compute001] GRES=dvd:2 SPEED=2200
NODECFG[compute002] GRES=dvd:2 SPEED=2200
NODECFG[compute003] GRES=dvd:2 SPEED=2200
NODECFG[compute004] GRES=dvd:2 SPEED=2200
NODECFG[compute005] SPEED=2200
NODECFG[compute006] SPEED=2200
NODECFG[compute007] SPEED=2200
NODECFG[compute008] SPEED=2200
```

```
# submit job which will allocate only from nodes 1 through 4 requesting one dvd per
task
> qsub -l nodes=2,walltime=100,gres=dvd job.cmd
```

*In this example, Moab determines that compute nodes exist that possess the requested generic resource. A compute node is a node object that possesses processors on which compute jobs actually execute. License server, network, and storage resources are typically represented by non-compute nodes. Because compute nodes exist with the requested generic resource, Moab interprets this job as requesting two compute nodes each of which must also possess a DVD generic resource.*

#### Example 3-141: Per Job Requests

```
NODECFG[network] PARTITION=shared GRES=bandwidth:2000000
```

```
# submit job which will allocate 2 nodes and 10000 units of network bandwidth
> qsub -l nodes=2,walltime=100,gres=bandwidth:10000 job.cmd
```

*In this example, Moab determines that there exist no compute nodes that also possess the generic resource `bandwidth` so this job is translated into a multiple-requirement—multi-req—job. Moab creates a job that has a requirement for two compute nodes and a second requirement for 10000 bandwidth generic resources. Because this is a multi-req job, Moab knows that it can locate these needed resources separately.*

## Using Generic Resource Requests in Conjunction with other Constraints

Jobs can explicitly specify generic resource constraints. However, if a job also specifies a [hostlist](#), the hostlist constraint overrides the generic resource constraint if the request is for per task allocation. In the Per Task Requests example, if the job also specified a hostlist, the `DVD` request is ignored.

## Requesting Resources with No Generic Resources

In some cases, it is valuable to allocate nodes that currently have no generic resources available. This can be done using the special value `none` as in the following example:

```
> qsub -l nodes=2,walltime=100,gres=none job.cmd
```

*In this case, the job only allocates compute nodes that have no generic resources associated with them.*

## Requesting Generic Resources Automatically within a Queue/Class

Generic resource constraints can be assigned to a queue or class and inherited by any jobs that do not have a **gres** request. This allows targeting of specific resources, automation of co-allocation requests, and other uses. To enable this, use the [DEFAULT.GRES](#) attribute of the [CLASSCFG](#) parameter as in the following example:

```
CLASSCFG[viz] DEFAULT.GRES=graphics:2
```

For each node requested by a *viz* job, also request two graphics cards.

## Managing Generic Resource Race Conditions

A software license race condition "window of opportunity" opens when Moab checks a license server for sufficient available licenses and closes when the user's software actually checks out the software licenses. The time between these two events can be seconds to many minutes depending on overhead factors such as node OS provisioning, job startup, licensed software startup, and so forth.

During this window, another Moab-scheduled job or a user or job external to the cluster or cloud can obtain enough software licenses that by the time the job attempts to obtain its software licenses, there are an insufficient quantity of available licenses. In such cases a job will sit and wait for the license, and while it waits it occupies but does not use resources that another job could have used. Use the **STARTDELAY** parameter to prevent such a situation.

```
GRESCFG[<license>] STARTDELAY=<window_of_opportunity>
```

With the **STARTDELAY** parameter enabled (on a per generic resource basis) Moab blocks any idle jobs requesting the same generic resource from starting until the *<window\_of\_opportunity>* passes. The window is defined by the customer on a per generic resource basis.

### Related topics

- [GRESCFG](#) parameter
- [Generic Metrics](#)
- [Generic Events](#)
- [General Node Attributes](#)
- [Floating Generic Resources](#)
- [Per Class Assignment of Generic Resource Consumption](#)
- [mnodectl -m](#) command to dynamically modify node resources
- [Favoring Jobs Based On Generic Resource Requirements](#)

## Enabling Generic Metrics

- [Configuring Generic Metrics](#)
- [Example Generic Metric Usage](#)

Moab allows organizations to enable generic performance metrics. These metrics allow decisions to be made and reports to be generated based on site specific environmental factors. This increases Moab's awareness of what is occurring within a given cluster environment, and allows arbitrary information to

be associated with resources and the workload within the cluster. Uses of these metrics are widespread and can cover anything from tracking node temperature, to memory faults, to application effectiveness.

- Execute triggers when specified thresholds are reached
- Modify node allocation affinity for specific jobs
- Initiate automated notifications when thresholds are reached
- Display current, average, maximum, and minimum metrics values in reports and charts within [Moab Cluster Manager](#)

## Configuring Generic Metrics

A new generic metric is automatically created and tracked at the server level if it is reported by either a node or a job.

To associate a generic metric with a job or node, a [native resource manager](#) must be set up and the `GMETRIC` attribute must be specified. For example, to associate a generic metric of `temp` with each node in a [TORQUE](#) cluster, the following could be reported by a native resource manager:

```
# temperature output
node001 GMETRIC[temp]=113
node002 GMETRIC[temp]=107
node003 GMETRIC[temp]=83
node004 GMETRIC[temp]=85
...
```



Generic metrics are tracked as floating point values allowing virtually any number to be reported.

In the preceding example, the new metric, `temp`, can now be used to monitor system usage and performance or to allow the scheduler to take action should certain thresholds be reached. Some uses include the following:

- Executing [triggers](#) based on generic metric thresholds
- Adjust a node's [availability](#) for accepting additional workload
- Adjust a node's [allocation priority](#)
- Initiate administrator [notification](#) of current, minimum, maximum, or average generic metric values
- Use metrics to report resource and job performance
- Use metrics to report resource and job failures
- Using job profiles to allow Moab to learn which resources best run which applications
- Tracking effective application efficiency to identify resource brown outseven when no node failure is obvious
- Viewing current and [historical](#) cluster-wide generic metric values to identify failure, performance, and usage
- Enable charging policies based on consumption of generic metrics patterns

- View changes in generic metrics on nodes, jobs, and cluster wide over time
- Submit jobs with generic metric based [node-allocation requirements](#)

Generic metric values can be viewed using [checkjob](#), [checknode](#), [mdiag -n,mdiag -j](#), or [Moab Cluster Manager](#) Charting and Reporting Features.



Historical job and node generic metric statistics can be cleared using the [mjobctl](#) and [mnodectl](#) commands.

## Example Generic Metric Usage

As an example, consider a cluster with two primary purposes for generic metrics. The first purpose is to track and adjust scheduling behavior based on node temperature to mitigate overheating nodes. The second purpose is to track and charge for utilization of a locally developed data staging service.

The first step in enabling a generic metric is to create probes to monitor and report this information. Depending on the environment, this information may be distributed or centralized. In the case of temperature monitoring, this information is often centralized by a hardware monitoring service and available via command line or an API. If monitoring a locally developed data staging service, this information may need to be collected from multiple remote nodes and aggregated to a central location. The following are popular freely available monitoring tools:

Tool	Link
<b>BigBrother</b>	<a href="http://www.bb4.org">http://www.bb4.org</a>
<b>Ganglia</b>	<a href="http://ganglia.sourceforge.net">http://ganglia.sourceforge.net</a>
<b>Monit</b>	<a href="http://www.tildeslash.com/monit">http://www.tildeslash.com/monit</a>
<b>Nagios</b>	<a href="http://www.nagios.org">http://www.nagios.org</a>

Once the needed probes are in place, a [native resource manager](#) interface must be created to report this information to Moab. Creating a native resource manager interface should be very simple, and in most cases a script similar to those found in the `$TOOLS_DIR($PREFIX/tools)` directory can be used as a template. For this example, we will assume centralized information and will use the RM script that follows:

```
#!/usr/bin/perl
# 'hwctl outputs information in format '<NODEID> <TEMP>'
open(TQUERY,"/usr/sbin/hwctl -q temp |");
while (<TQUERY>)
{
    my $nodeid,$temp = split /\w+/;
    $dstage=GetDSUsage($nodeid);
    print "$nodeid GMETRIC[temp]=$temp GMETRIC[dstage]=$dstage
";
}
```

With the script complete, the next step is to integrate this information into Moab. This is accomplished with the following configuration line:

```
RMCFG[local] TYPE=NATIVE CLUSTERQUERYURL=file://$TOOLS_DIR/node.query.local.pl
...
```

*Moab can now be recycled and temperature and data staging usage information will be integrated into Moab compute node reports.*

If the [checknode](#) command is run, output similar to the following is reported:

```
> checknode cluster013
...
Generic Metrics: temp=113.2,dstage=23748
...
```

*Moab Cluster Manager reports full current and historical generic metric information in its visual cluster overview screen.*

The next step in configuring Moab is to inform Moab to take certain actions based on the new information it is tracking. For this example, there are two purposes. The first purpose is to get jobs to avoid hot nodes when possible. This is accomplished using the **GMETRIC** attribute of the [Node Allocation Priority](#) function as in the following example:

```
NODEALLOCATIONPOLICY PRIORITY
NODECFG[DEFAULT] PRIORITYF=PRIORITY-10*GMETRIC[temp]
...
```

This simple priority function reduces the priority of the hottest nodes making such less likely to be allocated. See [Node Allocation Priority Factors](#) for a complete list of available priority factors.

The example cluster is also interested in notifying administrators if the temperature of a given node ever exceeds a critical threshold. This is accomplished using a [trigger](#). The following line will send email to administrators any time the temperature of a node exceeds 120 degrees.

```
NODECFG[DEFAULT] TRIGGER=atype=mail,etype=threshold,threshold=gmetric[temp]
>120,action='warning: node $OID temp high'
...
```

## Related topics

- [Simulation Overview](#)
- [Generic Consumable Resources](#)
- [Object Variables](#)
- [Generic Event Counters](#)

## Enabling Generic Events

- [Configuring Generic Events](#)
  - [Action Types](#)
  - [Named Events](#)

- [Generic Metric \(GMetric\) Events](#)
- [Reporting Generic Events](#)
  - [Using Generic Events for VM Detection](#)
- [Generic Events Attributes](#)
- [Manually Creating Generic Events](#)

Generic events are used to identify failures and other occurrences that Moab or other systems must be made aware. This information may result in automated resource recovery, notifications, adjustments to statistics, or changes in policy. Generic events also have the ability to carry an arbitrary human readable message that may be attached to associated objects or passed to administrators or external systems. Generic events typically signify the occurrence of a specific event as opposed to [generic metrics](#) which indicate a change in a measured value.

Using generic events, Moab can be configured to automatically address many failures and environmental changes improving the overall performance. Some sample events that sites may be interested in monitoring, recording, and taking action on include:

- Machine Room Status
  - Excessive Room Temperature
  - Power Failure or Power Fluctuation
  - Chiller Health
- Network File Server Status
  - Failed Network Connectivity
  - Server Hardware Failure
  - Full Network File System
- Compute Node Status
  - Machine Check Event (MCE)
  - Network Card (NIC) Failure
  - Excessive Motherboard/CPU Temperature
  - Hard Drive Failures

## Configuring Generic Events

Generic events are defined in the `moab.cfg` file and have several different configuration options. The only required option is **action**.

The full list of configurable options for generic events is contained in the following table:

Attribute	Description
<b>ACTION</b>	Comma-delimited list of actions to be processed when a new event is received.
<b>ECOUNT</b>	Number of events that must occur before launching action.  <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; background-color: #e6f2ff;">  Action will be launched each &lt;ECOUNT&gt; event if rearm is set.                 </div>
<b>REARM</b>	Minimum time between events specified in [ [ DD: ] HH: ] MM: ] SS format.
<b>SEVERITY</b>	An arbitrary severity level from 1 through 4, inclusive. SEVERITY appears in the output of <code>mdiag -n -v -v --xml</code> .  <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; background-color: #e6f2ff;">  The severity level will not be used for any other purpose.                 </div>

### Action Types

The impact of the event is controlled using the **ACTION** attribute of the **GEVENTCFG** parameter. The **ACTION** attribute is comma-delimited and may include any combination of the actions in the following table:

Value	Description
<b>DISABLE</b> [:<OTYPE>:<OID>]	Marks event object (or specified object) down until event report is cleared.
<b>EXECUTE</b>	Executes a script at the provided path. The value of <i>EXECUTE</i> is not contained in quotation marks. Arguments are allowed at the end of the path and are separated by question marks (?). <a href="#">Trigger variables</a> (such as <i>\$OID</i> ) are allowed.
<b>NOTIFY</b>	Notifies administrators of the event occurrence.
<b>OBJECTXMLSTDIN</b>	If the <i>EXECUTE</i> action type is also specified, this flag passes an XML description of the firing gevent to the script.
<b>OFF</b>	Powers off node or resource.
<b>ON</b>	Powers on node or resource.
<b>PREEMPT</b> [:<POLICY>]	Preempts workload associated with object (valid for node, job, reservation, partition, resource manager, user, group, account, class, QoS, and cluster objects).

Value	Description
<b>RECORD</b>	Records events to the event log. The record action causes a line to be added to the event log regardless of whether or not <a href="#">RECORDEVENTLIST</a> includes GEVENT.
<b>RESERVE</b> <b>[:&lt;DURATION&gt;]</b>	Reserves node for specified duration (default: 24 hours).
<b>RESET</b>	Resets object (valid for nodes - causes reboot).
<b>SIGNAL[:&lt;SIGNO&gt;]</b>	Sends signal to associated jobs or services (valid for node, job, reservation, partition, resource manager, user, group, account, class, QoS, and cluster objects).

This is an example of using `objectxmlstdin` with a `gevent`:

```
<gevent name="bob" statuscode="0" time="1320334763">Testing</gevent>
```

## Named Events

In general, generic events are named, with the exception of those based on [generic metrics](#). Names are used primarily to differentiate between different events and do not have any intrinsic meaning to Moab. It is suggested that the administrator choose names that denote specific meanings within the organization.

*Example 3-142:*

```
# Note: cpu failures require admin attention, create maintenance reservation
GEVENTCFG[cpufail] action=notify,record,disable,reserve rearm=01:00:00# Note: power
failures are transient, minimize future use
GEVENTCFG[powerfail] action=notify,record, rearm=00:05:00
# Note: fs full can be automatically fixed
GEVENTCFG[fsfull] action=notify,execute:/home/jason/MyPython/cleartmp.py?${OID}?nodefix
# Note: memory errors can cause invalid job results, clear node immediately
GEVENTCFG[badmem] action=notify,record,preempt,disable,reserve
```

## Generic Metric (GMetric) Events

GMetric events are generic events based on [generic metrics](#). They are used for executing an action when a generic metric passes a defined threshold. Unlike named events, GMetric events are not named and use the following format:

```
GEVENTCFG [GMETRIC<COMPARISON>VALUE] ACTION=...
```

*Example 3-143:*

```
GEVENTCFG [cputemp>150] action=off
```

This form of generic events uses the GMetric name, as returned by a **GMETRIC** attribute in a [native Resource Manager](#) interface.

**i** Only one generic event may be specified for any given generic metric.

Valid comparative operators are shown in the following table:

Type	Comparison	Notes
>	greater than	Numeric values only
> =	greater than or equal to	Numeric values only
= =	equal to	Numeric values only
<	less than	Numeric values only
< =	less than or equal to	Numeric values only
< >	not equal	Numeric values only

## Reporting Generic Events

Unlike [generic metrics](#), generic events can be optionally configured at the global level to adjust rearm policies, and other behaviors. In all cases, this is accomplished using the [GEVENTCFG](#) parameter.

To report an event associated with a job or node, use the [native Resource Manager](#) interface or the [mjobctl](#) or [mnodectl](#) commands. You can report generic events on the scheduler with the [mschedctl](#) command.

If using the native Resource Manager interface, use the [GEVENT](#) attribute as in the following example:

```
node001 GEVENT[hitemp]='temperature exceeds 150 degrees'
node017 GEVENT[fullfs]='/var/tmp is full'
```

**i** The time at which the event occurred can be passed to Moab to prevent multiple processing of the same event. This is accomplished by specifying the event type in the format `<GEVENTID> [:<EVENTTIME>]` as in what follows:

```
node001 GEVENT[hitemp:1130325993]='temperature exceeds 150 degrees'
node017 GEVENT[fullfs:1130325142]='/var/tmp is full'
```

## Using Generic Events for VM Detection

To enable Moab to detect a virtual machine (VM) reported by a generic event, do the following:

1. Set up your resource manager to detect virtual machine creation and to submit a generic event to Moab.
2. Configure `moab.cfg` to recognize a generic event.

```
GEVENTCFG[NewVM] ACTION=execute:/opt/moab/AddVM.py,OBJECTXMLSTDIN
```

3. Report the event.

```
> mschedctl -c gevent -n NewVM -m "VM=newVMName"
```

*With the `ObjectXMLStdin` action set, Moab sends an XML description of the generic event to the script, so the message passes through.*

The following sample Perl script submits a VMTracking job for the new VM:

```
#!/usr/bin/perl

# in moab.cfg: GEVENTCFG[NewVM] ACTION=execute:$TOOLSDDIR/newvm_event.pl,OBJECTXMLSTDIN
# trigger gevent with: mschedctl -c gevent -n NewVM -m "VM=TestVM1"
# input to this script: <gevent name="NewVM" statuscode="0"
time="1318500261">VM=TestVM1</gevent>

use strict;

my $vmidVarName = "preVMID";
my $vmTemplate = "existingVM";
my $vmOwner = "operator";

$ENV{MOABHOMEDIR} = '/opt/moab';

my $xml = join "", <STDIN>;
my ($vmid) = ($xml =~ m/VM=([^\<]+\</>)/);
if ( defined $vmid )
{
    my $cmd = qq| $ENV{MOABHOMEDIR}/bin/mvmctl -q $vmid --xml |;
    my $vmxml = ` $cmd `;
    my ($hv, $os, $proc, $disk, $mem) = (undef, undef, undef, undef, undef);
    ($hv) = ($vmxml =~ m/CONTAINERNODE="([^\"]+)/);
    ($os) = ($vmxml =~ m/OS="([^\"]+)/);
    ($proc) = ($vmxml =~ m/RCPROC="([^\"]+)/);
    ($mem) = ($vmxml =~ m/RCMEM="([^\"]+)/);
    ($disk) = ($vmxml =~ m/RCDISK="([^\"]+)/);
    die "Error parsing VM XML. Invalid VMID $vmid or $hv || $os || $proc || $mem ||
    $disk?
    "
        if ( ! defined $hv || !defined $os || !defined $proc || !defined $mem || !defined
    $disk );

    $cmd = qq| $ENV{MOABHOMEDIR}/bin/msub -l
    hostlist=$hv,os=$os,nodes=1:ppn=$proc,mem=$mem,file=$disk,template=$vmTemplate,VAR=$vm
    idVarName=$vmid --proxy=$vmOwner /dev/null |;
    my $msubout = ` $cmd `;
    die "Error executing msub. Output is:
    $msubout
    " if ( $? );
} else {
    die "Error parsing VMID from GEVENT message
    ";
}
```

## Generic Events Attributes

Each node will record the following about reported generic events:

- status - is event active
- message - human readable message associated with event
- count - number of event incidences reported since statistics were cleared
- time - time of most recent event

Each event can be individually cleared, annotated, or deleted by cluster administrators using a [mnodectl](#) command.

 Generic events are only available in Moab 4.5.0 and later.

## Manually Creating Generic Events

Generic events may be manually created on a physical node or VM.

To add GEVENT event with message "hello" to node02, do the following:

```
> mnodectl -m gevent=event:"hello" node02
```

To add GEVENT event with message "hello" to myvm, do the following:

```
> mvmctl -m gevent=event:"hello" myvm
```

### Related topics

- [Simulation Overview](#)
- [Generic Consumable Resources](#)
- [Object Variables](#)
- [Generic Event Counters](#)

# Resource Managers and Interfaces

- [Resource Manager Overview on page 585](#)
- [Resource Manager Configuration on page 588](#)
- [Resource Manager Extensions on page 618](#)
- [Adding New Resource Manager Interfaces on page 649](#)
- [Managing Resources Directly with the Native Interface on page 650](#)
- [Utilizing Multiple Resource Managers on page 661](#)
- [License Management on page 663](#)

- [Resource Provisioning on page 665](#)
- [Resource Manager Translation on page 672](#)

Moab provides a powerful resource management interface that enables significant flexibility in how resources and workloads are managed. Highlights of this interface are listed in what follows:

Highlight	Description
<b>Support for Multiple Standard Resource Manager Interface Protocols</b>	Manage cluster resources and workloads via PBS, Loadleveler, SGE, LSF, or BProc based resource managers.
<b>Support for Generic Resource Manager Interfaces</b>	Manage cluster resources securely via locally developed or open source projects using simple flat text interfaces or XML over HTTP.
<b>Support for Multiple Simultaneous Resource Managers</b>	Integrate resource and workload streams from multiple independent sources reporting disjoint sets of resources.
<b>Independent Workload and Resource Management</b>	Allow one system to manage your workload (queue manager) and another to manage your resources.
<b>Support for Rapid Development Interfaces</b>	Load resource and workload information directly from a file, a URL, or from the output of a configurable script or other executable.
<b>Resource Extension Information</b>	Integrate information from multiple sources to obtain a cohesive view of a compute resource. (That is, mix information from NIM, OpenPBS, FLEXlm, and a cluster performance monitor to obtain a single node image with a coordinated state and a more extensive list of node configuration and utilization attributes.)

## Resource Manager Overview

For most installations, the Moab Workload Manager uses the services of a resource manager to obtain information about the state of compute resources (nodes) and workload (jobs). Moab also uses the

resource manager to manage jobs, passing instructions regarding when, where, and how to start or otherwise manipulate jobs.

Moab can be configured to manage more than one resource manager simultaneously, even resource managers of different types. Using a local queue, jobs may even be migrated from one resource manager to another. However, there are currently limitations regarding jobs submitted directly to a resource manager (not to the local queue.) In such cases, the job is constrained to only run within the bound of the resource manager to which it was submitted.

- [Scheduler/Resource Manager Interactions](#)
  - [Resource Manager Commands](#)
  - [Resource Manager Flow](#)
- [Resource Manager Specific Details \(Limitations/Special Features\)](#)
- [Synchronizing Conflicting Information](#)
- [Evaluating Resource Manager Availability and Performance](#)

## Scheduler/Resource Manager Interactions

Moab interacts with all resource managers using a common set of commands and objects. Each resource manager interfaces, obtains, and translates Moab concepts regarding workload and resources into native resource manager objects, attributes, and commands.

Information on creating a new scheduler resource manager interface can be found in the [Adding New Resource Manager Interfaces](#) section.

### Resource Manager Commands

For many environments, Moab interaction with the resource manager is limited to the following objects and functions:

Object	Function	Details
Job	Query	Collect detailed state, requirement, and utilization information about jobs
	Modify	Change job state and/or attributes
	Start	Execute a job on a specified set of resources
	Cancel	Cancel an existing job
	Preempt/Resume	Suspend, resume, checkpoint, restart, or requeue a job

Object	Function	Details
Node	Query	Collect detailed state, configuration, and utilization information about compute resources
	Modify	Change node state and/or attributes
Queue	Query	Collect detailed policy and configuration information from the resource manager

Using these functions, Moab is able to fully manage workload, resources, and cluster policies. More detailed information about resource manager specific capabilities and limitations for each of these functions can be found in the individual resource manager overviews. (LL, PBS, LSF, SGE, BProc, or [WIKI](#)).

Beyond these base functions, other commands exist to support advanced features such as provisioning and cluster level resource management.

### Resource Manager Flow

In general, Moab interacts with resource managers in a sequence of steps each scheduling iteration. These steps are outlined in what follows:

1. load global resource information
2. load node specific information (optional)
3. load job information
4. load queue/policy information (optional)
5. cancel/preempt/modify jobs according to cluster policies
6. start jobs in accordance with available resources and policy constraints
7. handle user commands

Typically, each step completes before the next step is started. However, with current systems, size and complexity mandate a more advanced parallel approach providing benefits in the areas of reliability, concurrency, and responsiveness.

### Resource Manager Specific Details (Limitations/Special Features)

- TORQUE
  - [TORQUE Homepage](#)
- SLURM/Wiki
  - [SLURM Integration Guide](#)
  - [Wiki Overview](#)

## Synchronizing Conflicting Information

Moab does not trust resource manager information. Node, job, and policy information is reloaded on each iteration and discrepancies are detected. Synchronization issues and allocation conflicts are logged and handled where possible. To assist sites in minimizing stale information and conflicts, a number of policies and parameters are available.

- Node State Synchronization Policies (see [NODESYNCTIME](#) on page 1010)
- Stale Data Purging (see [JOBPURGETIME](#) on page 985)
- Thread Management (preventing resource manager failures from affecting scheduler operation)
- Resource Manager Poll Interval (see [RMPOLLINTERVAL](#) on page 1036)
- Node Query Refresh Rate (see [NODEPOLLFREQUENCY](#) on page 1006)

## Evaluating Resource Manager Availability and Performance

Each resource manager is individually tracked and evaluated by Moab. Using the [mdiag -R](#) on page 203 command, a site can determine how a resource manager is configured, how heavily it is loaded, what failures, if any, have occurred in the recent past, and how responsive it is to requests.

Related topics

- [Resource Manager Configuration](#)
- [Resource Manager Extensions](#)

## Resource Manager Configuration

- [Defining and Configuring Resource Manager Interfaces](#)
  - [Resource Manager Attributes](#)
- [Resource Manager Configuration Details](#)
  - [Resource Manager Types](#)
  - [Resource Manager Name](#)
  - [Resource Manager Location](#)
  - [Resource Manager Flags](#)
  - [Other Attributes](#)
- [Scheduler/Resource Manager Interactions](#)

## Defining and Configuring Resource Manager Interfaces

Moab resource manager interfaces are defined using the [RMCFG](#) on page 1035 parameter. This parameter allows specification of key aspects of the interface. In most cases, only the **TYPE** attribute needs to be specified and Moab determines the needed defaults required to activate and use the selected interface. In the following example, an interface to a Loadleveler resource manager is defined.



ADMINEXEC	
<b>Description</b>	Normally, when the <b>JOBSUBMITURL</b> is executed, Moab will drop to the UID and GID of the user submitting the job. Specifying an <b>ADMINEXEC</b> of <i>jobsubmit</i> causes Moab to use its own UID and GID instead (usually root). This is useful for some native resource managers where the <b>JOBSUBMITURL</b> is not a user command (such as <i>qsub</i> ) but a script that interfaces directly with the resource manager.
<b>Example</b>	<pre>RMCFG[base] ADMINEXEC=jobsubmit</pre> <p><i>Moab will not use the user's UID and GID for executing the <b>JOBSUBMITURL</b>.</i></p>

AUTHTYPE	
<b>Format</b>	One of <i>CHECKSUM</i> , <i>OTHER</i> , <i>PKI</i> , <i>SECUREPORT</i> , or <i>NONE</i> .
<b>Default</b>	<i>CHECKSUM</i>
<b>Description</b>	Specifies the security protocol to be used in scheduler-resource manager communication. <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> Only valid with WIKI based interfaces.</p> </div>
<b>Example</b>	<pre>RMCFG[base] AUTHTYPE=CHECKSUM</pre> <p><i>Moab requires a secret key-based checksum associated with each resource manager message.</i></p>

BANDWIDTH	
<b>Format:</b>	<FLOAT>[ {M G T} ]
<b>Default:</b>	-1 (unlimited)
<b>Description:</b>	Specifies the maximum deliverable bandwidth between the Moab server and the resource manager for staging jobs and data. Bandwidth is specified in units per second and defaults to a unit of MB/s. If a unit modifier is specified, the value is interpreted accordingly (M - megabytes/sec, G - gigabytes/sec, T - terabytes/sec).

## BANDWIDTH

## Example:

```
RMCFG[base] BANDWIDTH=340G
```

*Moab will reserve up to 340 GB of network bandwidth when scheduling job and data staging operations to and from this resource manager.*

## CHECKPOINTSIG

## Format

One of *suspend*, *<INTEGER>*, or *SIG<X>*

## Description

Specifies what signal to send the resource manager when a job is checkpointed (See [Checkpoint Overview](#)).

## Example

```
RMCFG[base] CHECKPOINTSIG=SIGKILL
```

*Moab routes the signal **SIGKILL** through the resource manager to the job when a job is checkpointed.*

## CHECKPOINTTIMEOUT

## Format

[ [ [DD:] HH:] MM:] SS

## Default

0 (no timeout)

## Description

Specifies how long Moab waits for a job to checkpoint before canceling it. If set to 0, Moab does not cancel the job if it fails to checkpoint (See [Checkpoint Overview](#)).

## Example

```
RMCFG[base] CHECKPOINTTIMEOUT=5:00
```

*Moab cancels any job that has not exited 5 minutes after receiving a checkpoint request.*

## CLIENT

## Format

<PEER>

## Default

Use name of resource manager for peer client lookup

CLIENT	
<b>Description</b>	If specified, the resource manager will use the peer value to authenticate remote connections. (See configuring peers). If not specified, the resource manager will search for a <a href="#">CLIENTCFG[&lt;X&gt;] on page 937</a> entry of RM: <RMNAME> in the moab-private.cfg file.
<b>Example</b>	<pre>RMCFG[clusterBI] CLIENT=clusterB</pre> <p>Moab will look up and use information for peer <i>clusterB</i> when authenticating the <i>clusterBI</i> resource manager.</p>

CLUSTERQUERYURL	
<b>Format</b>	<p>[file://&lt;path&gt;   http://&lt;address&gt;   &lt;path&gt;]</p> <p>If <i>file://</i> is specified, Moab treats the destination as a flat text file. If <i>http://</i> is specified, Moab treats the destination as a hypertext transfer protocol file. If just a path is specified, Moab treats the destination as an executable.</p>
<b>Description</b>	Specifies how Moab queries the resource manager (See <a href="#">Native RM</a> , <a href="#">URL Notes</a> , and <a href="#">interface details</a> ).
<b>Example</b>	<pre>RMCFG[base] CLUSTERQUERYURL=file:///tmp/cluster.config</pre> <p><i>Moab reads /tmp/cluster.config when it queries base resource manager.</i></p>

CONFIGFILE	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies the resource manager specific configuration file that must be used to enable correct API communication.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; background-color: #E6F2FF;"> <p> Only valid with LL- and SLURM-based interfaces.</p> </div>
<b>Example</b>	<pre>RMCFG[base] TYPE=LL CONFIGFILE=/home/loadl/loadl_config</pre> <p><i>The scheduler uses the specified file when establishing the resource manager/scheduler interface connection.</i></p>

DATARM	
<b>Format</b>	<RM NAME>
<b>Description</b>	If specified, the resource manager uses the given storage resource manager to handle staging data in and out.
<b>Example</b>	<pre>RMCFG[clusterB] DATARM=clusterB_storage</pre> <p><i>When data staging is required by jobs starting/completing on clusterB, Moab uses the storage interface defined by <b>clusterB_storage</b> to stage and monitor the data.</i></p>

DEFAULTCLASS	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the class to use if jobs submitted via this resource manager interface do not have an associated class.
<b>Example</b>	<pre>RMCFG[internal] DEFAULTCLASS=batch</pre> <p><i>Moab assigns the class <b>batch</b> to all jobs from the resource manager internal that do not have a class assigned.</i></p> <p><b>i</b> If you are using PBS as the resource manager, a job will never come from PBS without a class, and the default will never apply.</p>

DEFAULTHIGHSPEEDADAPTER	
<b>Format:</b>	<STRING>
<b>Default:</b>	<i>sn0</i>
<b>Description:</b>	Specifies the default high speed switch adapter to use when starting LoadLeveler jobs (supported in version 4.2.2 and higher of Moab and 3.2 of LoadLeveler).
<b>Example:</b>	<pre>RMCFG[base] DEFAULTHIGHSPEEDADAPTER=sn1</pre> <p><i>The scheduler will start jobs requesting a high speed adapter on sn1.</i></p>

DESCRIPTION	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the human-readable description for the resource manager interface. If white space is used, the description should be quoted.
<b>Example</b>	<pre>RMCFG[torque] DESCRIPTION='TORQUE RM for launching jobs'</pre> <p><i>Moab annotates the TORQUE resource manager accordingly.</i></p>

ENV	
<b>Format</b>	Semi-colon-delimited (;) list of <KEY>=<VALUE> pairs
<b>Default</b>	<i>MOABHOMEDIR=&lt;MOABHOMEDIR&gt;</i>
<b>Description</b>	Specifies a list of environment variables that will be passed to URLs of type <i>exec://</i> for that resource manager.
<b>Example</b>	<pre>RMCFG[base] ENV=HOST=node001;RETRYTIME=50 RMCFG[base] CLUSTERQUERYURL=exec:///opt/moab/tools/cluster.query.pl RMCFG[base] WORKLOADQUERYURL=exec:///opt/moab/tools/ workload.query.pl RMCFG[base] ENV=HOST=node001;RETRYTIME=50 RMCFG[base] CLUSTERQUERYURL=exec:///opt/moab/tools/cluster.query.pl RMCFG[base] WORKLOADQUERYURL=exec:///opt/moab/tools/workload.query.pl</pre> <p><i>The environment variables HOST and RETRYTIME (with values node001 and 50 respectively) are passed to the /opt/moab/tools/cluster.query.pl and /opt/moab/tools/workload.query.pl when they are executed.</i></p>

EPORT	
<b>Format:</b>	<INTEGER>
<b>Description:</b>	Specifies the event port to use to receive resource manager based scheduling events.
<b>Example:</b>	<pre>RMCFG[base] EPORT=15017</pre> <p><i>The scheduler will look for scheduling events from the resource manager host at port 15017.</i></p>

FAILTIME	
<b>Format:</b>	[[ [DD:]HH:]MM:]SS
<b>Description:</b>	Specifies how long a resource manager must be down before any failure triggers associated with the resource manager fire.
<b>Example:</b>	<pre>RMCFG[base] FAILTIME=3:00</pre> <p><i>If the base resource manager is down for three minutes, any resource manager failure triggers fire.</i></p>

FLAGS	
<b>Format</b>	Comma-delimited list of zero or more of the following: <a href="#">asyncdelete</a> , <a href="#">asyncstart</a> , <a href="#">autostart</a> , <a href="#">autosync</a> , <a href="#">client</a> , <a href="#">fullcp</a> , <a href="#">executionServer</a> , <a href="#">grid</a> , <a href="#">hostingCenter</a> , <a href="#">ignqueuestate</a> , <a href="#">private</a> , <a href="#">pushslavejobupdates</a> , <a href="#">report</a> , <a href="#">shared</a> , or <a href="#">static</a>
<b>Description</b>	Specifies various attributes of the resource manager. See <a href="#">Flag Details</a> for more information.
<b>Example</b>	<pre>RMCFG[base] FLAGS=static</pre> <p><i>Moab uses this resource manager to perform a single update of node and job objects reported elsewhere.</i></p>

FNLIST	
<b>Format</b>	Comma-delimited list of zero or more of the following: <a href="#">clusterquery</a> , <a href="#">jobcancel</a> , <a href="#">jobqueue</a> , <a href="#">jobresume</a> , <a href="#">jobstart</a> , <a href="#">jobsuspend</a> , <a href="#">queuequery</a> , <a href="#">resourcequery</a> or <a href="#">workloadquery</a>
<b>Description</b>	By default, a resource manager utilizes all functions supported to query and control batch objects. If this parameter is specified, only the listed functions are used.
<b>Example</b>	<pre>RMCFG[base] FNLIST=queuequery</pre> <p><i>Moab only uses this resource manager interface to load queue configuration information.</i></p>

HOST	
<b>Format</b>	<STRING>
<b>Default</b>	<i>localhost</i>
<b>Description</b>	The host name of the machine on which the resource manager server is running.
<b>Example</b>	<code>RMCFG[base] host=server1</code>

IGNHNODES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether to read in the PBSPro host nodes. This parameter is used in conjunction with <a href="#">USEVNODES on page 611</a> . When both are set to <i>TRUE</i> , the host nodes are not queried.
<b>Example</b>	<code>RMCFG[pbs] IGNHNODES=TRUE</code>

JOBCANCELURL	
<b>Format</b>	<protocol>:// [<host>[:<port>]] [<path>]
<b>Default</b>	---
<b>Description</b>	Specifies how Moab cancels jobs via the resource manager. (See <a href="#">URL Notes</a> below.)
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px;"> <code>RMCFG[base] JOBCANCELURL=exec:///opt/moab/job.cancel.lsf.pl</code> </div> <p><i>Moab executes /opt/moab/job.cancel.lsf.pl to cancel specific jobs.</i></p>

JOBEXTENDDURATION	
<b>Format</b>	[[ [DD:]HH:]MM:]SS [, [[ [DD:]HH:]MM:]SS [!] [<] (or <MIN TIME>[, <MAX TIME>] [!])

JOBEXTENDDURATION	
<b>Default</b>	---
<b>Description</b>	Specifies the minimum and maximum amount of time that can be added to a job's walltime if it is possible for the job to be extended. (See <a href="#">MINWCLIMIT</a> .) As the job runs longer than its current specified minimum wallclock limit (-l <i>minwclimit</i> , for example), Moab attempts to extend the job's limit by the minimum <b>JOBEXTENDDURATION</b> . This continues until either the extension can no longer occur (it is blocked by a reservation or job), the maximum <b>JOBEXTENDDURATION</b> is reached, or the user's specified wallclock limit (-l <i>walltime</i> ) is reached. When a job is extended, it is marked as <b>PREEMPTIBLE</b> , unless the ! is appended to the end of the configuration string. If the < is at the end of the string, however, the job is extended the maximum amount possible.
<b>Example</b>	<pre>RMCFG[base] JOBEXTENDDURATION=30,1:00:00</pre> <p><i>Moab extends a job's walltime by 30 seconds each time the job is about to run out of walltime until it is bound by one hour, a reservation/job, or the job's original "maximum" wallclock limit.</i></p>

JOBIDFORMAT	
<b>Format</b>	<i>INTEGER</i>
<b>Default</b>	---
<b>Description</b>	Specifies that Moab should use numbers to create job IDs. This eliminates multiple job IDs associated with a single job.
<b>Example</b>	<pre>RMCFG[base] JOBIDFORMAT=INTEGER</pre> <p><i>Job IDs are generated as numbers.</i></p>

JOBMODIFYURL	
<b>Format</b>	<protocol>:// [<host>[:<port>]] [<path>]
<b>Default</b>	---
<b>Description</b>	Specifies how Moab modifies jobs via the resource manager. (See <a href="#">URL Notes</a> , and <a href="#">interface details</a> .)

JOBMODIFYURL	
<b>Example</b>	<pre>RMCFG[base] JOBMODIFYURL=exec://\$TOOLSDIR/job.modify.dyn.pl</pre> <p><i>Moab executes /opt/moab/job.modify.dyn.pl to modify specific jobs.</i></p>

JOBRSVRECREATE	
<b>Format</b>	Boolean
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	Specifies whether Moab will re-create a job reservation each time job information is updated by a resource manager (See <a href="#">Considerations for Large Clusters</a> for more information.).
<b>Example</b>	<pre>RMCFG[base] JOBRSVRECREATE=FALSE</pre> <p><i>Moab only creates a job reservation once when the job first starts.</i></p>

JOBSTARTURL	
<b>Format</b>	<protocol>:// [<host>[:<port>]] [<path>]
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	Specifies how Moab starts jobs via the resource manager. (See <a href="#">URL Notes</a> below.)
<b>Example</b>	<pre>RMCFG[base] JOBSTARTURL=http://orion.bsu.edu:1322/moab/jobstart.cgi</pre> <p><i>Moab triggers the jobstart.cgi script via http to start specific jobs.</i></p>

JOBSUBMITURL	
<b>Format</b>	<protocol>:// [<host>[:<port>]] [<path>]
<b>Description</b>	Specifies how Moab submits jobs to the resource manager (See <a href="#">URL Notes</a> below.).

## JOBSUBMITURL

### Example

```
RMCFG[base] JOBSUBMITURL=exec://$TOOLS DIR/job.submit.dyn.pl
```

*Moab submits jobs directly to the database located on host  
dbserver.flc.com*

## JOBSUSPENDURL

### Format

<protocol>:// [<host>[:<port>]] [<path>]

### Description

Specifies how Moab suspends jobs via the resource manager. (See [URL Notes](#) below.)

### Example

```
RMCFG[base] JOBSUSPENDURL=EXEC://$HOME/scripts/job.suspend
```

*Moab executes the job.suspend script when jobs are suspended.*

## JOBVALIDATEURL

### Format

<protocol>:// [<host>[:<port>]] [<path>]

### Description

Specifies how Moab validates newly submitted jobs (See [URL Notes](#) below.). If the [script](#) returns with a non-zero exit code, the job is rejected. (See [User Proxying/Alternate Credentials](#).)

### Example

```
RMCFG[base] JOBVALIDATEURL=exec://$TOOLS/job.validate.pl
```

*Moab executes the 'job.validate.pl' script when jobs are submitted to verify they  
are acceptable.*

## MAXDSOP

### Format

<INTEGER>

### Default

-1 (unlimited)

### Description

Specifies the maximum number of data staging operations that may be simultaneously active.

### Example

```
RMCFG[ds] MAXDSOP=16
```

MAXITERATIONFAILURECOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	80
<b>Description</b>	Specifies the number of times the RM must fail within a certain iteration before Moab considers it down or corrupt. When an RM is down or corrupt, Moab will not attempt to interact with it.
<b>Example</b>	<pre>RMCFG [base] MAXITERATIONFAILURECOUNT=25</pre> <p><i>The RM base must fail 25 times in a single iteration for Moab to consider it down and cease interacting with it.</i></p>

MAXJOBPERMINUTE	
<b>Format</b>	<INTEGER>
<b>Default</b>	-1 (unlimited)
<b>Description</b>	Specifies the maximum number of jobs allowed to start per minute via the resource manager.
<b>Example</b>	<pre>RMCFG [base] MAXJOBPERMINUTE=5</pre> <p><i>The scheduler only allows five jobs per minute to launch via the resource manager base.</i></p>

MAXJOBS	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (limited only by the Moab <b>MAXJOB</b> setting)
<b>Description</b>	<p>Specifies the maximum number of active jobs that this interface is allowed to load from the resource manager.</p> <div style="border: 1px solid #0056b3; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> Only works with Moab peer resource managers at this time.</p> </div>

## MAXJOBS

## Example

```
RMCFG[cluster1] SERVER=moab://cluster1 MAXJOBS=200
```

*The scheduler loads up to 200 active jobs from the remote Moab peer cluster1.*

## MINETIME

## Format

<INTEGER>

## Default

1

## Description

Specifies the minimum time in seconds between processing subsequent scheduling events.

## Example

```
RMCFG[base] MINETIME=5
```

*The scheduler batch-processes scheduling events that occur less than five seconds apart.*

## NMPORT

## Format

<INTEGER>

## Default

(any valid port number)

## Description

Allows specification of the resource manager's node manager port and is only required when this port has been set to a non-default value.

## Example

```
RMCFG[base] NMPORT=13001
```

*The scheduler contacts the node manager located on each compute node at port 13001.*

## NODEFAILURERSVPROFILE

## Format

<STRING>

## Description

Specifies the rsv template to use when placing a reservation onto failed nodes (See also [NODEFAILURERESERVETIME](#) on page 1004.).

## NODEFAILURERSVPROFILE

## Example

```
# moab.cfg
RMCFG[base] NODEFAILURERSVPROFILE=long
RSVPROFILE[long] DURATION=25:00RSVPROFILE[long] USERLIST=john
```

*The scheduler will use the long rsv profile when creating reservations over failed nodes belonging to base.*

## NODESTATEPOLICY

## Format

One of *OPTIMISTIC* or *PESSIMISTIC*

## Default

*PESSIMISTIC*

## Description

Specifies how Moab should determine the state of a node when multiple resource managers are reporting state.

*OPTIMISTIC* specifies that if any resource manager reports a state of up, that state will be used. *PESSIMISTIC* specifies that if any resource manager reports a state of down, that state will be used.

## Example

```
# moab.cfg
RMCFG[native] TYPE=NATIVE NODESTATEPOLICY=OPTIMISTIC
```

## OMAP

## Format

<protocol>:// [<host>[:<port>]] [<path>]

## Description

Specifies an object map file that is used to map credentials and other objects when using this resource manager peer (See [Grid Credential Management](#) for full details).

## Example

```
moab.cfg
RMCFG[peer1] OMAP=file:///opt/moab/omap.dat
```

*When communicating with the resource manager peer1, objects are mapped according to the rules defined in the /opt/moab/omap.dat file.*

## PORT

## Format

<INTEGER>

PORT	
<b>Default</b>	0
<b>Description</b>	Specifies the port on which the scheduler should contact the associated resource manager. The value 0 specifies that the resource manager default port should be used.
<b>Example</b>	<pre>RMCFG[base] TYPE=PBS HOST=cws PORT=20001</pre> <p><i>Moab attempts to contact the PBS server daemon on host cws, port 20001.</i></p>

PROVDURATION	
<b>Format</b>	[ [ [DD:]HH:]MM:]SS
<b>Default</b>	2:30
<b>Description</b>	Specifies the upper bound (walltime) of a provisioning request. After this duration, Moab will consider the provisioning attempt failed.
<b>Example</b>	<pre>RMCFG[base] PROVDURATION=5:00</pre> <p><i>When RM base provisions a node for more than 5 minutes, Moab considers the provisioning as having failed.</i></p>

PTYSTRING	
<b>Format</b>	<STRING>
<b>Default</b>	<i>srun -n1 -N1 --pty</i>

## PTYSTRING

<p><b>Description</b></p>	<p>When a SLURM interactive job is submitted, it builds an <code>salloc</code> command that gets the requested resources and an <code>srun</code> command that creates a terminal session on one of the nodes. The <code>srun</code> command is called the <code>PTYString</code>. <code>PTYString</code> is configured in <code>moab.cfg</code>.</p> <p>There are two special things you can do with <code>PTYString</code>:</p> <ol style="list-style-type: none"> <li>1. You can have <code>PTYSTRING=\$salloc</code> which says to use the default <code>salloc</code> command (<code>SallocDefaultCommand</code>, look in the <code>slurm.conf</code> man page) defined in <code>slurm.conf</code>. Internally, Moab won't add a <code>PTYString</code> because SLURM will call the <code>SallocDefaultCommand</code>.</li> <li>2. As in the example below, you can add <code>\$SHELL</code>. <code>\$SHELL</code> will be expanded to either what you request on the command line (such as <code>msub -S /bin/tcsh -l</code>) or to the value of <code>\$SHELL</code> in your current session.</li> </ol> <p><code>PTYString</code> works only with SLURM.</p>
<p><b>Example</b></p>	<pre>RMCFG[slurm] PTYSTRING="srun -nl -N1 --pty --preserve-env \$SHELL"</pre>

## RESOURCECREATEURL

<p><b>Format</b></p>	<p>&lt;STRING&gt;</p>
<p><b>Default</b></p>	<p>[<i>exec</i>://&lt;path&gt;   <i>http</i>://&lt;address&gt;   &lt;path&gt;]</p> <p>If <i>exec</i>:// is specified, Moab treats the destination as an executable file; if <i>http</i>:// is specified, Moab treats the destination as a hypertext transfer protocol file.</p>
<p><b>Description</b></p>	<p>Specifies a script or method that can be used by Moab to create resources dynamically, such as creating a virtual machine on a hypervisor.</p>
<p><b>Example</b></p>	<pre>RMCFG[base] RESOURCECREATEURL=exec:///opt/script/vm.provision.py</pre> <p><i>Moab invokes the <code>vm.provision.py</code> script, passing in data as command line arguments, to request a creation of new resources.</i></p>

## RESOURCETYPE

<p><b>Format</b></p>	<p>{<i>COMPUTE</i> <i>FS</i> <i>LICENSE</i> <i>NETWORK</i> <i>PROV</i>}</p>
<p><b>Description</b></p>	<p>Specifies which type of resource this resource manager is configured to control. See <a href="#">Native Resource Managers</a> for more information.</p>

## RESOURCETYPE

### Example

```
RMCFG[base] TYPE=NATIVE RESOURCETYPE=FS
```

*Resource manager base will function as a **NATIVE** resource manager and control file systems.*

## RMSTARTURL

### Format

```
[exec://<path> | http://<address> | <path>]
```

If *exec://* is specified, Moab treats the destination as an executable file; if *http://* is specified, Moab treats the destination as a hypertext transfer protocol file.

### Description

Specifies how Moab starts the resource manager.

### Example

```
RMCFG[base] RMSTARTURL=exec:///tmp/nat.start.pl
```

*Moab executes /tmp/nat.start.pl to start the resource manager base.*

## RMSTOPURL

### Format

```
[exec://<path> | http://<address> | <path>]
```

If *exec://* is specified, Moab treats the destination as an executable file; if *http://* is specified, Moab treats the destination as a hypertext transfer protocol file.

### Description

Specifies how Moab stops the resource manager.

### Example

```
RMCFG[base] RMSTOPURL=exec:///tmp/nat.stop.pl
```

*Moab executes /tmp/nat.stop.pl to stop the resource manager base.*

## SBINDIR

### Format

```
<PATH>
```

### Description

For use with TORQUE; specifies the location of the TORQUE system binaries (supported in TORQUE 1.2.0p4 and higher).

SBINDIR	
<b>Example</b>	<pre>RMCFG[base] TYPE=pbs SBINDIR=/usr/local/torque/sbin</pre> <p><i>Moab tells TORQUE that its system binaries are located in /usr/local/torque/sbin.</i></p>
SERVER	
<b>Format</b>	<URL>
<b>Description</b>	Specifies the resource management service to use. If not specified, the scheduler locates the resource manager via built-in defaults or, if available, with an information service.
<b>Example</b>	<pre>RMCFG[base] server=ll://supercluster.org:9705</pre> <p><i>Moab attempts to use the Loadleveler scheduling API at the specified location.</i></p>
SLURMFLAGS	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies characteristics of the SLURM resource manager interface. The <b>COMPRESSOUTPUT</b> flag instructs Moab to use the compact hostlist format for job submissions to SLURM. The flag <b>NODEDELTAQUERY</b> instructs Moab to request delta node updates when it queries SLURM for node configuration.
<b>Example</b>	<pre>RMCFG[slurm] SLURMFLAGS=COMPRESSOUTPUT</pre> <p><i>Moab uses the <b>COMPRESSOUTPUT</b> flag to determine interface characteristics with SLURM.</i></p>
SOFTTERMSIG	
<b>Format</b>	<INTEGER>or SIG<X>
<b>Description</b>	Specifies what signal to send the resource manager when a job reaches its soft wallclock limit. (See <a href="#">JOBMAXOVERRUN</a> .)

## SOFTTERMSIG

## Example

```
RMCFG[base] SOFTTERMSIG=SIGUSR1
```

*Moab routes the signal **SIGUSR1** through the resource manager to the job when a job reaches its soft wallclock limit.*

## STAGETHRESHOLD

## Format

```
[ [DD:]HH:]MM:]SS
```

## Description

Specifies the maximum time a job waits to start locally before considering being migrated to a remote peer. In other words, if a job's start time on a remote cluster is less than the start time on the local cluster, but the difference between the two is less than **STAGETHRESHOLD**, then the job is scheduled locally. The aim is to avoid job/data staging overhead if the difference in start times is minimal.



If this attribute is used, backfill is disabled for the associated resource manager.

## Example

```
RMCFG[remote_cluster] STAGETHRESHOLD=00:05:00
```

*Moab only migrates jobs to remote\_cluster if the jobs can start five minutes sooner on the remote cluster than they could on the local cluster.*

## STARTCMD

## Format

```
<STRING>
```

## Description

Specifies the full path to the resource manager job start client. If the resource manager API fails, Moab executes the specified start command in a second attempt to start the job.



Moab calls the start command with the format `<CMD><JOBID> -H <HOSTLIST>` unless the environment variable `MOABNOHOSTLIST` is set in which case Moab will only pass the job ID.

## Example

```
RMCFG[base] STARTCMD=/usr/local/bin/qrun
```

*Moab uses the specified start command if API failures occur when launching jobs.*

SUBMITCMD	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the full path to the resource manager job submission client.
<b>Example</b>	<pre>RMCFG[base] SUBMITCMD=/usr/local/bin/qsub</pre> <p><i>Moab uses the specified submit command when migrating jobs.</i></p>

SUBMITPOLICY	
<b>Format</b>	One of <i>NODECENTRIC</i> or <i>PROCCENTRIC</i>
<b>Default</b>	<i>PROCCENTRIC</i>
<b>Description</b>	If set to <i>NODECENTRIC</i> , each specified node requested by the job is interpreted as a true compute host, not as a task or processor.
<b>Example</b>	<pre>RMCFG[base] SUBMITPOLICY=NODECENTRIC</pre> <p><i>Moab uses the specified submit policy when migrating jobs.</i></p>

SUSPENDSIG	
<b>Format</b>	<INTEGER> (valid UNIX signal between 1 and 64)
<b>Default</b>	RM-specific default
<b>Description</b>	If set, Moab sends the specified signal to a job when a job suspend request is issued.
<b>Example</b>	<pre>RMCFG[base] SUSPENDSIG=19</pre> <p><i>Moab uses the specified suspend signal when suspending jobs within the base resource manager.</i></p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> <b>SUSPENDSIG</b> should not be used with <a href="#">TORQUE</a> or other PBS-based resource managers.</p> </div>

SYNCJOBID	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Specifies that Moab should migrate jobs to the local resource manager with the job's Moab-assigned job ID. In a grid, the grid-head will only pass dependencies to the underlying Moab if <b>SYNCJOBID</b> is set. This attribute can be used with the <a href="#">JOBIDFORMAT on page 597</a> attribute and <a href="#">PROXYJOBSUBMISSION on page 617</a> flag in order to synchronize job IDs between Moab and the resource manager. For more information about all steps necessary to synchronize job IDs between Moab and TORQUE, see <a href="#">Synchronizing Job IDs in TORQUE and Moab on page 613</a> .
<b>Example</b>	<pre>RMCFG[slurm] TYPE=wiki:slurm SYNCJOBID=TRUE</pre>

SYSTEMMODIFYURL	
<b>Format</b>	<pre>[exec://&lt;path&gt;   http://&lt;address&gt;   &lt;path&gt;]</pre> If <code>exec://</code> is specified, Moab treats the destination as an executable file; if <code>http://</code> is specified, Moab treats the destination as a hypertext transfer protocol file.
<b>Description</b>	Specifies how Moab modifies attributes of the system. This interface is used in <a href="#">data staging</a> .
<b>Example</b>	<pre>RMCFG[base] SYSTEMMODIFYURL=exec:///tmp/system.modify.pl</pre> <i>Moab executes /tmp/system.modify.pl when it modifies system attributes in conjunction with the resource manager base.</i>

SYSTEMQUERYURL	
<b>Format</b>	<pre>[exec://&lt;path&gt;   http://&lt;address&gt;   &lt;path&gt;]</pre> If <code>file://</code> is specified, Moab treats the destination as a flat text file; if <code>http://</code> is specified, Moab treats the destination as a hypertext transfer protocol file; if just a path is specified, Moab treats the destination as an executable.
<b>Description</b>	Specifies how Moab queries attributes of the system. This interface is used in <a href="#">data staging</a> .
<b>Example</b>	<pre>RMCFG[base] SYSTEMQUERYURL=file:///tmp/system.query</pre> <i>Moab reads /tmp/system.query when it queries the system in conjunction with base resource manager.</i>

TARGETUSAGE	
<b>Format</b>	<INTEGER>[%]
<b>Default</b>	90%
<b>Description</b>	Amount of resource manager resources to explicitly use. In the case of a storage resource manager, indicates the target usage of data storage resources to dedicate to active data migration requests. If the specified value contains a percent sign (%), the target value is a percent of the configured value. Otherwise, the target value is considered to be an absolute value measured in megabytes (MB).
<b>Example</b>	<pre>RMCFG[storage] TYPE=NATIVE RESOURCETYPE=storage RMCFG[storage] TARGETUSAGE=80%</pre> <p><i>Moab schedules data migration requests to never exceed 80% usage of the storage resource manager's disk cache and network resources.</i></p>

TIMEOUT	
<b>Format</b>	<INTEGER>
<b>Default</b>	30
<b>Description</b>	Time (in seconds) the scheduler waits for a response from the resource manager.
<b>Example</b>	<pre>RMCFG[base] TIMEOUT=40</pre> <p><i>Moab waits 40 seconds to receive a response from the resource manager before timing out and giving up. Moab tries again on the next iteration.</i></p>

TRIGGER	
<b>Format</b>	<TRIG_SPEC>
<b>Description</b>	A <a href="#">trigger</a> specification indicating behaviors to enforce in the event of certain events associated with the resource manager, including resource manager start, stop, and failure.
<b>Example</b>	<pre>RMCFG[base] TRIGGER=&lt;X&gt;</pre>

TYPE	
<b>Format</b>	<RMTYPE>[:<RMSUBTYPE>] where <RMTYPE> is one of the following: <a href="#">TORQUE</a> , <a href="#">NATIVE</a> , <a href="#">PBS</a> , <a href="#">RMS</a> , <a href="#">SSS</a> , or <a href="#">WIKI</a> and the optional <RMSUBTYPE> value is one of RMS.
<b>Default</b>	PBS
<b>Description</b>	<p>Specifies type of resource manager to be contacted by the scheduler.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p><b>i</b> For TYPE WIKI, AUTHTYPE must be set to CHECKSUM. The &lt;RMSUBTYPE&gt; option is currently only used to support Compaq's RMS resource manager in conjunction with PBS. In this case, the value <code>PBS:RMS</code> should be specified.</p> </div>
<b>Example</b>	<pre>RMCFG[clusterA] TYPE=PBS HOST=clusterA PORT=15003 RMCFG[clusterB] TYPE=PBS HOST=clusterB PORT=15005</pre> <p><i>Moab interfaces to two different PBS resource managers, one located on server clusterA at port 15003 and one located on server clusterB at port 15005.</i></p>

USEVNODES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether to schedule on PBS virtual nodes. When set to <i>TRUE</i> , Moab queries PBSPro for vnodes and puts jobs on vnodes rather than hosts. In some systems, such as PBS + Altix, it may not be desirable to read in the host nodes; for such situations refer to the <a href="#">IGNHNODES</a> attribute.
<b>Example</b>	<pre>RMCFG[pbs] USEVNODES=TRUE</pre>

VARIABLES	
<b>Format</b>	<VAR>=<VAL>[, <VAR>=<VAL>]
<b>Description</b>	Opaque resource manager variables.
<b>Example</b>	<pre>RMCFG[base] VARIABLES=SCHEDDHOST=head1</pre> <p><i>Moab associates the variable <code>SCHEDDHOST</code> with the value <code>head1</code> on resource manager base.</i></p>

VERSION	
<b>Format</b>	<STRING>
<b>Default</b>	<i>SLURM: 10200</i> (i.e., 1.2.0)
<b>Description</b>	Resource manager-specific version string.
<b>Example</b>	<pre>RMCFG[base] VERSION=10124</pre> <p><i>Moab assumes that resource manager base has a version number of 1.1.24.</i></p>

VMOWNERRM	
<b>Format</b>	<STRING>
<b>Description</b>	Used with provisioning resource managers that can create VMs. It specifies the resource manager that will own any VMs created by the resource manager.
<b>Example</b>	<pre>RMCFG[torque] RMCFG[prov] RESOURCETYPE=PROV VMOWNERRM=torque</pre>

WORKLOADQUERYURL	
<b>Format</b>	<p><i>[file://&lt;path&gt;   http://&lt;address&gt;   &lt;path&gt;]</i></p> <p>If <i>file://</i> is specified, Moab treats the destination as a flat text file; if <i>http://</i> is specified, Moab treats the destination as a hypertext transfer protocol file; if just a path is specified, Moab treats the destination as an executable.</p>
<b>Description</b>	Specifies how Moab queries the resource manager for workload information. (See <a href="#">Native RM</a> , <a href="#">URL Notes</a> , and <a href="#">interface details</a> .)
<b>Example</b>	<pre>RMCFG[TORQUE] WORKLOADQUERYURL=exec://\$TOOLSDIR/job.query.dyn.pl</pre> <p><i>Moab executes /opt/moab/tools/job.query.dyn.pl to obtain updated workload information from resource manager TORQUE.</i></p>

URL notes

URL parameters can load files by using the *file*, *exec*, and *http* protocols.

For the protocol *file*, Moab loads the data directly from the text file pointed to by path.

```
RMCFG[base] SYSTEMQUERYURL=file:///tmp/system.query
```

For the protocol *exec*, Moab executes the file pointed to by path and loads the output written to STDOUT. If the script requires arguments, you can use a question mark (?) between the script name and the arguments, and an ampersand (&) for each space.

```
RMCFG[base] JOBVALIDATEURL=exec://$TOOLS/job.validate.pl
RMCFG[native] CLUSTERQUERYURL=exec://opt/moab/tools/cluster.query.pl?-group=group1&-arch=x86
```

### Synchronizing Job IDs in TORQUE and Moab

**i** Unless you use an [msub on page 286](#) submit filter or you're in a grid, it is recommended that you use your RM-specific job submission command (for instance, `qsub`).

In order to synchronize your job IDs between TORQUE and Moab you must perform the following steps:

1. Verify that you are using TORQUE version 2.5.6 or later.
2. Set [SYNCJOBID on page 609](#) to *TRUE* in all resource managers.

```
RMCFG[torque] TYPE=PBS SYNCJOBID=TRUE
```

3. Set the [PROXYJOB SUBMISSION on page 617](#) flag. With *PROXYJOB SUBMISSION* enabled, you must run Moab as a TORQUE manager or operator. Verify that other users can submit jobs using [msub on page 286](#). Moab, as a non-root user, should still be able to submit jobs to TORQUE and synchronize job IDs.

```
RMCFG[torque] TYPE=PBS SYNCJOBID=TRUE
RMCFG[torque] FLAGS=PROXYJOB SUBMISSION
```

4. Add [JOBIDFORMAT on page 597](#)=*INTEGER* to the internal RM. Adding this parameter forces Moab to only use numbers as job IDs and those numbers to synchronize across Moab, TORQUE, and the entire grid. This enhances the end-user experience as it eliminates multiple job IDs associated with a single job.

```
RMCFG[torque] TYPE=PBS SYNCJOBID=TRUE
RMCFG[torque] FLAGS=PROXYJOB SUBMISSION

RMCFG[internal] JOBIDFORMAT=INTEGER
```

## Resource Manager Configuration Details

As with all scheduler parameters, follows the syntax described within the [Parameters Overview](#).

### Resource Manager Types

The **RMCFG** parameter allows the scheduler to interface to multiple types of resource managers using the **TYPE** or **SERVER** attributes. Specifying these attributes, any of the following listed resource managers may be supported.

Type	Resource managers	Details
<b>Moab</b>	Moab Workload Manager	Use the Moab peer-to-peer (grid) capabilities to enable grids and other configurations. (See <a href="#">Grid Configuration</a> .)
<b>MWS</b>	Moab Web Services	The MWS resource manager type is a native integration between Moab and MWS. Resource manager data is passed directly between Moab and MWS using JSON (rather than Moab's native WIKI syntax). This simplifies RM configuration for systems where one or more MWS plugins are acting as resource managers. See the "Moab Workload Manager resource manager integration" section of the MWS plugins chapter in the MWS documentation for more information.
<b>Native</b>	Moab <i>Native</i> Interface	Used for connecting directly to scripts, files, and databases. (See <a href="#">Managing Resources Directly with the Native Interface</a> .)
<b>PBS</b>	TORQUE (all versions)	N/A
<b>SSS</b>	Scalable Systems Software Project version 2.0 and higher	N/A
<b>WIKI</b>	<a href="#">Wiki</a> interface specification version 1.0 and higher	Used for LRM, YRM, ClubMASK, BProc, SLURM, and others.

## Resource Manager Name

Moab can support more than one resource manager simultaneously. Consequently, the **RMCFG** parameter takes an index value such as `RMCFG[clusterA]`. This index value essentially names the resource manager (as done by the deprecated parameter **RMNAME**). The resource manager name is used by the scheduler in diagnostic displays, logging, and in reporting resource consumption to the allocation manager. For most environments, the selection of the resource manager name can be arbitrary.

## Resource Manager Location

The **HOST**, **PORT**, and **SERVER** attributes can be used to specify how the resource manager should be contacted. For many resource managers the interface correctly establishes contact using default values. These parameters need only to be specified for resource managers such as the WIKI interface (that do

not include defaults) or with resources managers that can be configured to run at non-standard locations (such as PBS). In all other cases, the resource manager is automatically located.

## Resource Manager Flags

The **FLAGS** attribute can be used to modify many aspects of a resources manager's behavior.

Flag	Description
<b>ASYNCSTART</b>	Jobs started on this resource manager start asynchronously. In this case, the scheduler does not wait for confirmation that the job correctly starts before proceeding. (See <a href="#">Large Cluster Tuning</a> for more information.)
<b>AUTOSTART</b>	Jobs staged to this resource manager do not need to be explicitly started by the scheduler. The resource manager itself handles job launch.
<b>AUTOSYNC</b>	Resource manager starts and stops together with Moab. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> This requires that the resource manager support a resource manager start and stop API or the <a href="#">RMSTARTURL</a> and <a href="#">RMSTOPURL</a> attributes are set.</p> </div>
<b>BECOMEMASTER</b>	Nodes reported by this resource manager will transfer ownership to this resource manager if they are currently owned by another resource manager that does not have this flag set.
<b>CLIENT</b>	A client resource manager object is created for diagnostic/statistical purposes or to configure Moab's interaction with this resource manager. It represents an external entity that consumes server resources or services, allows a local administrator to track this usage, and configures specific policies related to that resource manager. A client resource manager object loads no data and provides no services.
<b>CLOCKSKEWCHECKING</b>	Setting <a href="#">CLOCKSKEWCHECKING</a> allows you to configure clock skew adjustments. Most of the time it is sufficient to use an NTP server to keep the clocks in your system synchronized.
<a href="#">COLLAPSEDVIEW</a>	Does not work — not supported The resource manager masks details about local workload and resources and presents only information relevant to the remote server.
<b>DYNAMICCRED</b>	The resource manager creates credentials within the cluster as needed to support workload. (See <a href="#">Identity Manager Overview</a> .)
<b>EXECUTIONSERVER</b>	The resource manager is capable of launching and executing batch workload.

Flag	Description
<b>FSISREMOTE</b>	Add this flag if the working file system doesn't exist on the server to prevent Moab from validating files and directories at migration.
<b>FULLCP</b>	Always checkpoint full job information (useful with <a href="#">Native</a> resource managers).
<b>HOSTINGCENTER</b>	The resource manager interface is used to negotiate an adjustment in dynamic resource access.
<b>IGNQUEUESTATE</b>	The queue state reported by the resource manager should be ignored. May be used if queues must be disabled inside of a particular resource manager to allow an external scheduler to properly operate.
<b>IGNWORKLOADSTATE</b>	When this flag is applied to a native resource manager, any jobs that are reported via that resource manager's "workload query URL" have their reported state ignored. For example, if an RM has the <i>IgnWorkloadState</i> flag and it reports that a set of jobs have a state of "Running," this state is ignored and the jobs will either have a default state set or will inherit the state from another RM reporting on that same set of jobs. This flag only changes the behavior of RMs of type <i>NATIVE</i> .
<b>LOCALWORKLOADEXPORT</b>	When set, destination peers share information about local and remote jobs, allowing job management of different clusters at a single peer. For more information, see <a href="#">Workload Submission and Control</a> .
<b>MIGRATEALLJOBATTRIBUTES</b>	When set, this flag causes additional job information to be migrated to the resource manager; additional job information includes things such as node features applied via <code>CLASSCFG[name] DEFAULT.FEATURES</code> , the account to which the job was submitted, and job walltime limit.
<b>NOAUTORES</b>	If the resource manager does not report CPU usage to Moab because CPU usage is at 0%, Moab assumes full CPU usage. When set, Moab recognizes the resource manager report as 0% usage. This is only valid for PBS.
<b>NOCREATERESOURCE</b>	To use resources discovered from this resource manager, they must be created by another resource manager first. For example, if you set <i>NOCREATERESOURCE</i> on RM A, which reports nodes 1 and 2, and RM B only reports node 1, then node 2 will not be created because RM B did not report it.
<b>PRIVATE</b>	The resources and workload reported by the resource manager are not reported to non-administrator users.

Flag	Description
<b>PROXYJOB SUBMISSION</b>	Enables Admin proxy job submission, which means administrators may submit jobs in behalf of other users.
<b>PUSHSLAVEJOBUPDATES</b>	Enables job changes made on a grid slave to be pushed to the grid head or master. Without this flag, jobs being reported to the grid head do not show any changes made on the remote Moab server (via <a href="#">mjobctl</a> and so forth).
<b>RECORDGPUMETRICS</b>	Enables the recording of GPU metrics for nodes.
<b>RECORDMICMETRICS</b>	Enables the recording of MIC metrics for nodes.
<b>REPORT</b>	N/A
<b>SHARED</b>	Resources of this resource manager may be scheduled by multiple independent sources and may not be assumed to be owned by any single source.
<b>STATIC</b>	This resource manager only provides partial object information and this information does not change over time. Consequently, this resource manager may only be called once per object to modify job and node information.
<b>USERSPACEISSEPARATE</b>	This tells Moab to ignore validating the user's uid and gid in the case that information doesn't exist on the Moab server.

### Example

```
# resource manager 'torque' should use asynchronous job start
# and report resources in 'grid' mode
RMCFG[torque] FLAGS=asyncstart,grid
```

## Scheduler/Resource Manager Interactions

In the simplest configuration, Moab interacts with the resource manager using the following four primary functions:

Function	Description
<b>GETJOBINFO</b>	Collect detailed state and requirement information about idle, running, and recently completed jobs.
<b>GETNODEINFO</b>	Collect detailed state information about idle, busy, and defined nodes.

Function	Description
<b>STARTJOB</b>	Immediately start a specific job on a particular set of nodes.
<b>CANCELJOB</b>	Immediately cancel a specific job regardless of job state.

Using these four simple commands, Moab enables nearly its entire suite of scheduling functions. More detailed information about resource manager specific requirements and semantics for each of these commands can be found in the specific resource manager (such as [WIKI](#)) overviews.

In addition to these base commands, other commands are required to support advanced features such as suspend/resume, gang scheduling, and scheduler initiated checkpoint restart.

Information on creating a new scheduler resource manager interface can be found in the [Adding New Resource Manager Interfaces](#) section.

## Resource Manager Extensions

- [Resource Manager Extension Specification](#)
- [Resource Manager Extension Values](#)
- [Resource Manager Extension Examples](#)

All resource managers are not created equal. There is a wide range in what capabilities are available from system to system. Additionally, there is a large body of functionality that many, if not all, resource managers have no concept of. A good example of this is job QoS. Since most resource managers do not have a concept of quality of service, they do not provide a mechanism for users to specify this information. In many cases, Moab is able to add capabilities at a global level. However, a number of features require a *per job* specification. Resource manager extensions allow this information to be associated with the job.

### Resource Manager Extension Specification

Specifying resource manager extensions varies by resource manager. TORQUE, OpenPBS, PBSPro, Loadleveler, LSF, S3, and Wiki each allow the specification of an *extension* field as described in the following table:

Resource manager	Specification method
<b>TORQUE 2.0+</b>	-l <pre>&gt; qsub -l nodes=3,qos=high sleepy.cmd</pre>

Resource manager	Specification method
<b>TORQUE 1.x/OpenPBS</b>	<p><code>-W x=</code></p> <pre>&gt; qsub -l nodes=3 -W x=qos:high sleepy.cmd</pre> <p><b>i</b> OpenPBS does not support this ability by default but can be patched as described in the <a href="#">PBS Resource Manager Extension Overview</a>.</p>
<b>Loadleveler</b>	<p><code>#@comment</code></p> <pre>#@nodes = 3 #@comment = qos:high</pre>
<b>LSF</b>	<p><code>-ext</code></p> <pre>&gt; bsub -ext advres:system.2</pre>
<b>PBSPro</b>	<p><code>-l</code></p> <pre>&gt; qsub -l advres=system.2</pre> <p><b>i</b> Use of PBSPro resources requires configuring the <code>server_priv/resourcedef</code> file to define the needed extensions as in the following example:</p> <pre>advres type=string qos type=string sid type=string sjid type=string</pre>
<a href="#">Wiki</a>	<p><code>comment</code></p> <pre>comment=qos:high</pre>

## Resource Manager Extension Values

Using the resource manager specific method, the following job extensions are currently available:

[ADVRES on page 620](#)  
[BANDWIDTH on page 620](#)  
[CPUCLOCK on page 621](#)  
[DDISK on page 623](#)  
[DEADLINE on page 623](#)  
[DEPEND on page 624](#)  
[DMEM on page 624](#)  
[EPILOGUE on page 624](#)  
[EXCLUDENODES on page 625](#)  
[FEATURE on page 625](#)  
[GATTR on page 625](#)  
[GEOMETRY on page 626](#)  
[GMETRIC on page 626](#)  
[GPUs on page 626](#)  
[GRES](#)  
[HOSTLIST on page 628](#)  
[JGROUP on page 629](#)  
[JOBFLAGS](#)  
[JOBREJECTPOLICY on page 630](#)  
[MAXMEM on page 630](#)

[MAXPROC on page 631](#)  
[MEM on page 631](#)  
[MICS on page 631](#)  
[MINPREEMPTTIME on page 632](#)  
[MINPROCSPEED on page 632](#)  
[MINWCLIMIT on page 633](#)  
[MSTAGEIN on page 633](#)  
[MSTAGEOUT on page 634](#)  
[NACCESSPOLICY on page 635](#)  
[NALLOCPOLICY on page 636](#)  
[NCPUS on page 636](#)  
[NMATCHPOLICY on page 637](#)  
[NODESET on page 637](#)  
[NODESETCOUNT on page 637](#)  
[NODESETDELAY on page 637](#)  
[NODESETISOPTIONAL on page 638](#)  
[OPSYS on page 638](#)  
[PARTITION on page 638](#)  
[PLACEMENT on page 638](#)  
[PMEM on page 639](#)

[PROCS on page 639](#)  
[PROLOGUE on page 640](#)  
[PVMEM on page 640](#)  
[QoS on page 641](#)  
[QUEUEJOB on page 641](#)  
[REQATTR on page 641](#)  
[RESFAILPOLICY on page 642](#)  
[RMTYPE on page 642](#)  
[SIGNAL on page 642](#)  
[SOFTWARE](#)  
[SPRIORITY on page 642](#)  
[TEMPLATE on page 643](#)  
[TERMTIME on page 643](#)  
[TPN on page 643](#)  
[TRIG on page 644](#)  
[TRL](#)  
[VAR on page 645](#)  
[VC on page 646](#)  
[VMEM on page 646](#)

## ADVRES

### Format

[!] <RSVID>

### Description

Specifies that reserved resources are required to run the job. If <RSVID> is specified, then only resources within the specified reservation may be allocated (see [Job to Reservation Binding](#)). You can request to not use a specific reservation by using `advres=!<reservationname>`.

### Example

```
> qsub -l advres=grid.3
```

*Resources for the job must come from grid.3.*

```
> qsub -l advres=!grid.5
```

*Resources for the job must not come from grid.5*

## BANDWIDTH

### Format

<DOUBLE> (in MB/s)

### Description

Minimum available network bandwidth across allocated resources (See [Network Management](#)).

### Example

```
> bsub -ext bandwidth=120 chemjob.txt
```

CPUCLOCK	
<b>Format</b>	<STRING>

## CPUCLOCK

## Description

Specify the CPU clock frequency for each node requested for this job. A **cpuclock** request applies to every processor on every node in the request. Specifying varying CPU frequencies for different nodes or different processors on nodes in a single job request is not supported.

Not all CPUs support all possible frequencies or ACPI states. If the requested frequency is not supported by the CPU, the nearest frequency is used.

Using **cpuclock** sets **NODEACCESSPOLICY** to **SINGLEJOB**.

ALPS 1.4 or later is required when using **cpuclock** on Cray.

The clock frequency can be specified via:

- a number that indicates the clock frequency (with or without the SI unit suffix).
- a Linux power governor policy name. The governor names are:
  - **performance**: This governor instructs Linux to operate each logical processor at its maximum clock frequency.  
This setting consumes the most power and workload executes at the fastest possible speed.
  - **powersave**: This governor instructs Linux to operate each logical processor at its minimum clock frequency.  
This setting executes workload at the slowest possible speed. This setting does not necessarily consume the least amount of power since applications execute slower, and may actually consume more energy because of the additional time needed to complete the workload's execution.
  - **ondemand**: This governor dynamically switches the logical processor's clock frequency to the maximum value when system load is high and to the minimum value when the system load is low.  
This setting causes workload to execute at the fastest possible speed or the slowest possible speed, depending on OS load. The system switches between consuming the most power and the least power.



The power saving benefits of **ondemand** might be non-existent due to frequency switching latency if the system load causes clock frequency changes too often.

This has been true for older processors since changing the clock frequency required putting the processor into the C3 "sleep" state, changing its clock frequency, and then waking it up, all of which required a significant amount of time.

Newer processors, such as the Intel Xeon E5-2600 Sandy Bridge processors, can change clock frequency dynamically and much faster.

- **conservative**: This governor operates like the **ondemand** governor but is more conservative in switching between frequencies. It switches more gradually and uses all possible clock frequencies.  
This governor can switch to an intermediate clock frequency if it seems appropriate to the system load and usage, which the **ondemand** governor does not do.
- an ACPI performance state (or P-state) with or without the P prefix. P-states are a special

CPUCLOCK	
	<p>range of values (0-15) that map to specific frequencies. Not all processors support all 16 states, however, they all start at P0. P0 sets the CPU clock frequency to the highest performance state which runs at the maximum frequency. P15 sets the CPU clock frequency to the lowest performance state which runs at the lowest frequency.</p> <p>When reviewing job or node properties when <b>cpuclock</b> was used, be mindful of unit conversion. The OS reports frequency in Hz, not MHz or GHz.</p>
<b>Example</b>	<pre>msub -l cpuclock=1800,nodes=2 script.sh msub -l cpuclock=1800mhz,nodes=2 script.sh</pre> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to 1800 MHz.</i></p> <pre>msub -l cpuclock=performance,nodes=2 script.sh</pre> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to the performance power governor policy.</i></p> <pre>msub -l cpuclock=3,nodes=2 script.sh msub -l cpuclock=p3,nodes=2 script.sh</pre> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to a performance state of 3.</i></p>

DDISK	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Dedicated disk per task in MB.
<b>Example</b>	<pre>&gt; qsub -l ddisk=2000</pre>

DEADLINE	
<b>Format</b>	<p>Relative time: [ [ [DD:] HH:] MM:] SS</p> <p>Absolute time: hh:mm:ss_mm/dd/yy</p>
<b>Description</b>	Either the relative completion deadline of job (from job submission time) or an absolute deadline in which you specify the date and time the job will finish.

DEADLINE	
<b>Example:</b>	<pre>&gt; qsub -l deadline=2:00:00,nodes=4 /tmp/bio3.cmd</pre> <p><i>The job's deadline is 2 hours after its submission.</i></p>

DEPEND	
<b>Format</b>	[<DEPENDTYPE>:] [{jobname jobid}.] <ID>[: [{jobname jobid}.] <ID>]...
<b>Description</b>	Allows specification of job dependencies for compute or system jobs. If no ID prefix (jobname or jobid) is specified, the ID value is interpreted as a job ID.
<b>Example</b>	<pre># submit job which will run after job 1301 and 1304 complete &gt; msub -l depend=orion.1301:orion.1304 test.cmd orion.1322 # submit jobname-based dependency job &gt; msub -l depend=jobname.data1005 dataet1.cmd orion.1428</pre>

DMEM	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Dedicated memory per task in bytes.
<b>Example</b>	<pre>&gt; msub -l dmem=20480</pre> <p><i>Moab will dedicate 20 MB of memory to the task.</i></p>

EPILOGUE	
<b>Format</b>	<STRING>

EPILOGUE	
<b>Description</b>	<p>Specifies a user owned epilogue script which is run before the system epilogue and <code>epilogue.user</code> scripts at the completion of a job. The syntax is <code>epilogue=&lt;file&gt;</code>. The file can be designated with an absolute or relative path.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> This parameter works only with TORQUE.</p> </div>
<b>Example</b>	<pre>&gt; msub -l epilogue=epilogue_script.sh job.sh</pre>

EXCLUDENODES	
<b>Format</b>	<code>{&lt;nodeid&gt; &lt;node_range&gt;}[:...]</code>
<b>Description</b>	Specifies nodes that should not be considered for the given job.
<b>Example</b>	<pre>&gt; msub -l excludenodes=k1:k2:k[5-8] # Comma separated ranges work only with SLURM &gt; msub -l excludenodes=k[1-2,5-8]</pre>

FEATURE	
<b>Format</b>	<code>&lt;FEATURE&gt;[{: }&lt;FEATURE&gt;]...</code>
<b>Description</b>	<p>Required list of node attribute/node <a href="#">features</a>.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> If the <i>pipe</i> ( ) character is used as a delimiter, the features are logically OR'd together and the associated job may use resources that match any of the specified features.</p> </div>
<b>Example</b>	<pre>&gt; qsub -l feature='fastos:bigio' testjob.cmd</pre>

GATTR	
<b>Format</b>	<code>&lt;STRING&gt;</code>
<b>Description</b>	Generic job attribute associated with job. The maximum size for an attribute is 63 bytes (the core Moab size limit of 64, including a null byte)
<b>Example</b>	<pre>&gt; qsub -l gattr=bigjob</pre>

GEOMETRY	
<b>Format:</b>	{ (<TASKID>[, <TASKID>[, ...]]) [ (<TASKID>[, ...]) ... ] }
<b>Description:</b>	Explicitly specified task geometry.
<b>Example:</b>	<pre>&gt; qsub -l nodes=2:ppn=4 -W x=geometry:'{(0,1,4,5) (2,3,6,7)}' quanta2.cmd</pre> <p><i>The job quanta2.cmd runs tasks 0, 1, 4, and 5 on one node, while tasks 2, 3, 6, and 7 run on another node.</i></p>

GMETRIC	
<b>Format</b>	Generic metric requirement for allocated nodes where the requirement is specified using the format <GMNAME>[:{lt:, le:, eq:, ge:, gt:, ne:}<VALUE>]
<b>Description</b>	Indicates generic constraints that must be found on all allocated nodes. If a <VALUE> is not specified, the node must simply possess the generic metric (See <a href="#">Generic Metrics</a> for more information.).
<b>Example</b>	<pre>&gt; qsub -l gmetric=bioversion:ge:133244 testj.txt</pre>

GPUs	
<b>Format</b>	<pre>msub -l nodes=&lt;VALUE&gt;:ppn=&lt;VALUE&gt;:gpus=&lt;VALUE&gt;[:mode] [:reseterr]</pre> <p>Where mode is one of:</p> <ul style="list-style-type: none"> <li><i>exclusive</i> - The default setting. The GPU is used exclusively by one process thread.</li> <li><i>exclusive_thread</i> - The GPU is used exclusively by one process thread.</li> <li><i>exclusive_process</i> - The GPU is used exclusively by one process regardless of process thread.</li> </ul> <p>If present, <code>reseterr</code> resets the ECC memory bit error counters. This only resets the volatile error counts, or errors since the last reboot. The permanent error counts are not affected.</p> <p>Moab passes the <code>mode</code> and <code>reseterr</code> portion of the request to TORQUE for processing.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> Moab does not support requesting GPUs as a GRES. Submitting <code>msub -l gres=gpus:x</code> does not work.</p> </div>
<b>Description</b>	Moab schedules GPUs as a special type of <a href="#">node-locked generic resources</a> . When <a href="#">TORQUE reports GPUs</a> to Moab, Moab can schedule jobs and correctly assign GPUs to ensure that jobs are scheduled efficiently. To have Moab schedule GPUs, configure them in TORQUE then submit jobs using the "GPU" attribute. Moab automatically parses the "GPU" attribute and assigns them in the correct manner. For information about GPU metrics, see <a href="#">GPGPUMetrics</a> .

## GPUs

## Examples

```
> msub -l nodes=2:ppn=2:gpus=1:exclusive_process:reseterr
```

*Submits a job that requests 2 tasks, 2 processors and 1 GPU per task (2 GPUs total). Each GPU runs only threads related to the task and resets the volatile ECC memory big error counts at job start time.*

```
> msub -l nodes=4:gpus=1,tpn=2
```

*Submits a job that requests 4 tasks, 1 GPU per node (4 GPUs total), and 2 tasks per node. Each GPU is dedicated exclusively to one task process and the ECC memory bit error counters are not reset.*

```
> msub -l nodes=4:gpus=1:reseterr
```

*Submits a job that requests 4 tasks, 1 processor and 1 GPU per task (4 GPUs total). Each GPU is dedicated exclusively to one task process and resets the volatile ECC memory bit error counts at job start time.*

```
> msub -l nodes=4:gpus=2+1:ppn=2,walltime=600
```

*Submits a job that requests two different types of tasks, the first is 4 tasks, each with 1 processor and 2 gpus, and the second is 1 task with 2 processors. Each GPU is dedicated exclusively to one task process and the ECC memory bit error counters are not reset.*

## GRES and SOFTWARE

## Format

Percent sign (%) delimited list of generic resources where each resource is specified using the format `<RESTYPE>[+|:]<COUNT>`

## Description

Indicates generic resources required by the job. If the generic resource is node-locked, it is a per-task count. If a `<COUNT>` is not specified, the resource count defaults to 1.

## Example

```
> qsub -W x=GRES:tape+2%matlab+3 testj.txt
```



When specifying more than one generic resource with -l, use the percent (%) character to delimit them.

```
> qsub -l gres=tape+2%matlab+3 testj.txt
> qsub -l software=matlab:2 testj.txt
```

HOSTLIST	
<b>Format</b>	Comma (,) or plus (+) delimited list of hostnames. Ranges and regular expressions are supported in <code>msub</code> only.
<b>Description</b>	<p>Indicates an <i>exact set</i>, <i>superset</i>, or <i>subset</i> of nodes on which the job must run. Use the caret (^) or asterisk (*) characters to specify a host list as <i>superset</i> or <i>subset</i> respectively.</p> <p>An exact set is defined without a caret or asterisk. An exact set means <i>all</i> the hosts in the specified hostlist must be selected for the job.</p> <p>A subset means the specified hostlist is used first to select hosts for the job. If the job requires more hosts than are in the subset hostlist, they will be obtained from elsewhere if possible. If the job does not require all of the nodes in the subset hostlist, it will use only the ones it needs.</p> <p>A superset means the hostlist is the <i>only</i> source of hosts that should be considered for running the job. If the job can't find the necessary resources in the superset hostlist it should <i>not</i> run. No other hosts should be considered in allocating the job.</p>

## HOSTLIST

## Examples

```
> msub -l hostlist=nodeA+nodeB+nodeE
```

```
hostlist=foo[1-5]
```

*This is an exact set of (foo1,foo2,...,foo5). The job must run on all these nodes.*

```
hostlist=foo1+foo[3-9]
```

*This is an exact set of (foo1,foo3,foo4,...,foo9). The job must run on all these nodes.*

```
hostlist=foo[1,3-9]
```

*This is an exact set of the same nodes as the previous example.*

```
hostlist=foo[1-3]+bar[72-79]
```

*This is an exact set of (foo1,foo2,foo3,bar72,bar73,...,bar79). The job must run on all these nodes.*

```
hostlist=^node[1-50]
```

*This is a superset of (node1,node2,...,node50). These are the only nodes that can be considered for the job. If the necessary resources for the job are not in this hostlist, the job is not run. If the job does not require all the nodes in this hostlist, it will use only the ones that it needs.*

```
hostlist=*node[15-25]
```

*This is a subset of (node15,node16,...,node25). The nodes in this hostlist are considered first for the job. If the necessary resources for the job are not in this hostlist, Moab tries to obtain the necessary resources from elsewhere. If the job does not require all the nodes in this hostlist, it will use only the ones that it needs.*

## JGROUP

## Format

```
<JOBGROUPID>
```

## Description

ID of job group to which this job belongs (different from the GID of the user running the job).

## Example

```
> msub -l JGROUP=bluegroup
```

JOBFLAGS (aka FLAGS)	
<b>Format</b>	One or more of the following colon delimited job flags including ADVRES[:RSVID], NOQUEUE, NORMSTART, PREEMPTEE, PREEMPTOR, RESTARTABLE, or SUSPENDABLE (see <a href="#">job flag overview</a> for a complete listing).
<b>Description</b>	Associates various flags with the job.
<b>Example</b>	<pre>&gt; qsub -l nodes=1,walltime=3600,jobflags=advres myjob.py</pre>

JOBREJECTPOLICY	
<b>Format:</b>	One or more of <i>CANCEL</i> , <i>HOLD</i> , <i>IGNORE</i> (beta), <i>MAIL</i> , or <i>RETRY</i>
<b>Default:</b>	<i>HOLD</i>
<b>Details:</b>	<p>Specifies the action to take when the scheduler determines that a job can never run. <i>CANCEL</i> issues a call to the resource manager to cancel the job. <i>HOLD</i> places a batch hold on the job preventing the job from being further evaluated until released by an administrator.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin: 5px 0;"> <p> Administrators can dynamically alter job attributes and possibly <i>fix</i> the job with <a href="#">mjobctl -m</a>.</p> </div> <p>With <i>IGNORE</i> (currently in beta), the scheduler will allow the job to exist within the resource manager queue but will neither process it nor report it. <a href="#">MAIL</a> will send email to both the admin and the user when rejected jobs are detected. If <i>RETRY</i> is set, then Moab will allow the job to remain idle and will only attempt to start the job when the policy violation is resolved. Any combination of attributes may be specified. See <a href="#">QOSREJECTPOLICY</a>.</p> <p>This is a per-job policy specified with <a href="#">msub -l JOBREJECTPOLICY</a> also exists as a global parameter.</p>
<b>Example:</b>	<pre>&gt; msub -l jobrejectpolicy=cancel:mail</pre>

MAXMEM	
<b>Forma:</b>	<INTEGER> (in megabytes)
<b>Description</b>	Maximum amount of memory the job may consume across all tasks before the <a href="#">JOBMEM</a> action is taken.
<b>Example</b>	<pre>&gt; qsub -W x=MAXMEM:1000mb bw.cmd</pre> <div style="border: 1px dashed #000; border-radius: 5px; padding: 5px; margin: 5px 0;"> <p>If a <a href="#">RESOURCELIMITPOLICY</a> is set for per-job memory utilization, its action will be taken when this value is reached.</p> </div>

MAXPROC	
<b>Format</b>	<INTEGER>
<b>Description</b>	Maximum CPU load the job may consume across all tasks before the <a href="#">JOBPROC</a> action is taken.
<b>Example</b>	<pre>&gt; qsub -W x=MAXPROC:4 bw.cmd</pre> <p><i>If a <a href="#">RESOURCELIMITPOLICY</a> is set for per-job processor utilization, its action will be taken when this value is reached.</i></p>

MEM	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specify the maximum amount of physical memory used by the job. If you do not specify MB or GB, Moab uses bytes if your resource manager is TORQUE and MB if your resource manager is Native.
<b>Example</b>	<pre>&gt; msub -l nodes=4:ppn=2,mem=1024mb</pre> <p><i>The job must have 4 compute nodes with 2 processors per node. The job is limited to 1024 MB of memory.</i></p>

MICs	
<b>Format</b>	<pre>msub -l nodes=&lt;VALUE&gt;;ppn=&lt;VALUE&gt;;mics=&lt;VALUE&gt;[:mode]</pre> <p>Where mode is one of:</p> <ul style="list-style-type: none"> <li><i>exclusive</i> - The default setting. The MIC is used exclusively by one process thread.</li> <li><i>exclusive_thread</i> - The MIC is used exclusively by one process thread.</li> <li><i>exclusive_process</i> - The MIC is used exclusively by one process regardless of process thread.</li> </ul> <p>Moab passes the mode portion of the request to TORQUE for processing.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> Moab does not support requesting MICs as a GRES. Submitting <code>msub -l gres=mics:x</code> does not work.</p> </div>
<b>Description</b>	Moab schedules MICs as a special type of <a href="#">node-locked generic resources</a> . When <a href="#">TORQUE reports MICs</a> to Moab, Moab can schedule jobs and correctly assign MICs to ensure that jobs are scheduled efficiently. To have Moab schedule MICs, configure them in TORQUE then submit jobs using the "MIC" attribute. Moab automatically parses the "MIC" attribute and assigns them in the correct manner.

MICs	
<b>Examples</b>	<pre>&gt; msub -l nodes=2:ppn=2:mics=1:exclusive_process</pre> <p><i>Submits a job that requests 2 tasks, 2 processors and 1 MIC per task (2 MICs total). Each MIC runs only threads related to the task.</i></p>
	<pre>&gt; msub -l nodes=4:mics=1,tpn=2</pre> <p><i>Submits a job that requests 4 tasks, 1 MIC per node (4 MICs total), and 2 tasks per node. Each MIC is dedicated exclusively to one task process.</i></p>
	<pre>&gt; msub -l nodes=4:mics=1</pre> <p><i>Submits a job that requests 4 tasks, 1 processor and 1 MIC per task (4 MICs total). Each MIC is dedicated exclusively to one task process.</i></p>
	<pre>&gt; msub -l nodes=4:mics=2+1:ppn=2,walltime=600</pre> <p><i>Submits a job that requests two different types of tasks, the first is 4 tasks, each with 1 processor and 2 MICs, and the second is 1 task with 2 processors. Each MIC is dedicated exclusively to one task process.</i></p>

MINPREEMPTTIME	
<b>Format</b>	[ [DD:]HH:]MM:]SS
<b>Description</b>	<p>Minimum time job must run before being eligible for preemption.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> Can only be specified if associated <a href="#">QoS</a> allows per-job preemption configuration by setting the <a href="#">preemptconfig</a> flag.</p> </div>
<b>Example</b>	<pre>&gt; qsub -l minpreempttime=900 bw.cmd</pre> <p><i>Job cannot be preempted until it has run for 15 minutes.</i></p>

MINPROCSPEED	
<b>Format</b>	<INTEGER>
<b>Default</b>	0

MINPROCSPEED	
<b>Description</b>	Minimum <a href="#">processor speed</a> (in MHz) for every node that this job will run on.
<b>Example</b>	<pre>&gt; qsub -W x=MINPROCSPEED:2000 bw.cmd</pre> <p><i>Every node that runs this job must have a processor speed of at least 2000 MHz.</i></p>

MINWCLIMIT	
<b>Format</b>	[ [DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	Minimum wallclock limit job must run before being eligible for extension (See <a href="#">JOBEXTENDDURATION</a> or <a href="#">JOBEXTENDSTARTWALLTIME</a> ).
<b>Example</b>	<pre>&gt; qsub -l minwclimit=300,walltime=16000 bw.cmd</pre> <p><i>Job will run for at least 300 seconds but up to 16,000 seconds if possible (without interfering with other jobs).</i></p>

MSTAGEIN	
<b>Format</b>	[<SRCURL>[ <SRCRUL>... ]%]<DSTURL>

## MSTAGEIN

**Descrip-  
tion**

Indicates a job has [data staging](#) requirements. The source URL(s) listed will be transferred to the execution system for use by the job. If more than one source URL is specified, the destination URL must be a directory.

The format of `<SRCURL>` is: `[PROTO://] [HOST] [:PORT] [/PATH]` where the path is local.

The format of `<DSTURL>` is:  
`[PROTO://] [HOST] [:PORT] [/PATH]` where the path is remote.

PROTO can be any of the following protocols: ssh, file, or gsiftp.

HOST is the name of the host where the file resides.

PATH is the path of the source or destination file. The destination path may be a directory when sending a single file and must be a directory when sending multiple files. If a directory is specified, it must end with a forward slash (/).

Valid variables include:

`$JOBID`

`$HOME` - Path the script was run from

`$RHOME` - Home dir of the user on the remote system

`$SUBMITHOST`

`$DEST` - This is the Moab where the job will run

`$LOCALDATASTAGEHEAD`



If no destination is given, the protocol and file name will be set to the same as the source.



The `$RHOME` (remote home directory) variable is for when a user's home directory on the compute node is different than on the submission host.

**Examp-  
le:**

```
> msub -
Wx='mstagein=file://$HOME/helperscript.sh|file:///home/dev/datafile.txt%ssh://host/hom
e/dev/' script.sh
```

Copy `helperscript.sh` and `datafile.txt` from the local machine to `/home/dev/` on host for use in execution of `script.sh`. `$HOME` is a path containing a preceding / (i.e. `/home/adaptive`)

## MSTAGEOUT

**Forma-  
t**

`[<SRCURL>[|<SRCURL>... ]%]<DSTURL>`

## MSTAGEOUT

**Descri-  
ption**

Indicates whether a job has [data staging](#) requirements. The source URL(s) listed will be transferred from the execution system after the completion of the job. If more than one source URL is specified, the destination URL must be a directory.

The format of `<SRCURL>` is: `[PROTO://] [HOST] [:PORT] [/PATH]` where the path is remote.

The format of `<DSTURL>` is: `[PROTO://] [HOST] [:PORT] [/PATH]` where the path is local.

PROTO can be any of the following protocols: ssh, file, or gsiftp.

HOST is the name of the host where the file resides.

PATH is the path of the source or destination file. The destination path may be a directory when sending a single file and must be a directory when sending multiple files. If a directory is specified, it must end with a forward slash (/).

Valid variables include:

`$JOBID`

`$HOME` - Path the script was run from

`$RHOME` - Home dir of the user on the remote system

`$SUBMITHOST`

`$DEST` - This is the Moab where the job will run

`$LOCALDATASTAGEHEAD`

 If no destination is given, the protocol and file name will be set to the same as the source.

 The `$RHOME` (remote home directory) variable is for when a user's home directory on the compute node is different than on the submission host.

**Examp-  
le**

```
> msub -W
x='mstageout=ssh://$DEST/$HOME/resultfile1.txt|ssh://host/home/dev/resultsript.sh%file
: //home/dev/' script.sh
```

*Copy resultfile1.txt and resultsript.sh from the execution system to /home/dev/ after the execution of script.sh is complete. \$HOME is a path containing a preceding / (i.e. /home/adaptive).*

## NACCESSPOLICY

**Format**

One of [SHARED](#), [SINGLEJOB](#), [SINGLETASK](#), [SINGLEUSER](#), or [UNIQUEUSER](#)

NACCESSPOLICY	
<b>Description</b>	<p>Specifies how node resources should be accessed. (See <a href="#">Node Access Policies</a> for more information).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> The <b>naccesspolicy</b> option can only be used to make node access more constraining than is specified by the system, partition, or node policies. If the effective node access policy is <b>shared</b>, <b>naccesspolicy</b> can be set to <b>singleuser</b>, if the effective node access policy is <b>singlejob</b>, <b>naccesspolicy</b> can be set to <b>singletask</b>.</p> </div>
<b>Example</b>	<pre>&gt; qsub -l naccesspolicy=singleuser bw.cmd</pre> <pre>&gt; bsub -ext naccesspolicy=singleuser lancer.cmd</pre> <p><i>Job can only allocate free nodes or nodes running jobs by same user.</i></p>

NALLOCPOLICY	
<b>Format</b>	One of the valid settings for the parameter <a href="#">NODEALLOCATIONPOLICY</a>
<b>Description</b>	Specifies how node resources should be selected and allocated to the job. (See <a href="#">Node Allocation Policies</a> for more information.)
<b>Example</b>	<pre>&gt; qsub -l nallocpolicy=minresource bw.cmd</pre> <p><i>Job should use the <b>minresource</b> node allocation policy.</i></p>

NCPUS	
<b>Format</b>	<INTEGER>
<b>Description</b>	<p>The number of processors in one task where a task cannot span nodes. If <b>NCPUS</b> is used, then the resource manager's <a href="#">SUBMITPOLICY</a> should be set to <b>NODECENTRIC</b> to get correct behavior. <code>-l ncpus=&lt;#&gt;</code> is equivalent to <code>-l nodes=1 :ppn=&lt;#&gt;</code> when <b>JOBNODEMATCHPOLICY</b> is set to <b>EXACTNODE</b>. <b>NCPUS</b> is used when submitting jobs to an SMP. When using <a href="#">GPUs</a> to submit to an SMP, use <code>-l ncpus=&lt;#&gt;:GPUs=&lt;#&gt;</code>.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> You cannot request both <b>ncpus</b> and <b>nodes</b> in the same job.</p> </div>

NMATCHPOLICY	
<b>Format</b>	One of the valid settings for the parameter <a href="#">JOBNODEMATCHPOLICY</a>
<b>Description</b>	Specifies how node resources should be selected and allocated to the job.
<b>Example</b>	<pre>&gt; qsub -l nodes=2 -W x=nmatchpolicy:exactnode bw.cmd</pre> <p><i>Job should use the EXACTNODEJOBNODEMATCHPOLICY.</i></p>

NODESET	
<b>Format</b>	<SETTYPE>:<SETATTR>[:<SETLIST>]
<b>Description</b>	Specifies nodeset constraints for job resource allocation (See the <a href="#">NodeSet Overview</a> for more information.).
<b>Example</b>	<pre>&gt; qsub -l nodeset=ONEOF:FEATURE:fastos:hiprio:bigmem bw.cmd</pre>

NODESETCOUNT	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies how many node sets a job uses.
<b>Example</b>	<pre>&gt; msub -l nodesetcount=2</pre>

NODESETDELAY	
<b>Format</b>	[ [ [DD:]HH:]MM:]SS
<b>Description</b>	Causes Moab to attempt to span a job evenly across nodesets unless doing so delays the job beyond the requested <b>NODESETDELAY</b> .
<b>Example</b>	<pre>&gt; qsub -l nodesetdelay=300,walltime=16000 bw.cmd</pre>

NODESETISOPTIONAL	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Specifies whether the nodeset constraint is optional (See the <a href="#">NodeSet Overview</a> for more information).
	 Requires SCHEDCFG[] <a href="#">FLAGS</a> =allowperjobnodesetisoptional.
<b>Example</b>	<pre>&gt; msub -l nodesetisoptional=true bw.cmd</pre>

OPSYS	
<b>Format</b>	<OperatingSystem>
<b>Description</b>	Specifies the job's required operating system.
<b>Example</b>	<pre>&gt; qsub -l nodes=1,opsys=rh73 chem92.cmd</pre>

PARTITION	
<b>Format</b>	<STRING>[:<STRING>]...
<b>Description</b>	Specifies the <a href="#">partition</a> (or partitions) in which the job must run.
	 The job must have access to this partition based on system wide or credential based partition access lists.
<b>Example</b>	<pre>&gt; qsub -l nodes=1,partition=math:geology</pre> <i>The job must only run in the <b>math</b> partition or the <b>geology</b> partition.</i>

PLACEMENT	
<b>Format</b>	[numa=X] [[:]sockets=Y] [[:]usetreads]
<b>Description</b>	Specifies the task placement of jobs.

PLACEMENT	
<b>Example</b>	<pre>&gt; msub -l nodes=4:ppn=2,placement=numa=2</pre> <p><i>This means to place the job on 4 compute nodes with 2 processors per node, with 2 different NUMA nodes per compute node, and 1 processor per NUMA node.</i></p>
PMEM	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies the maximum amount of physical memory used by any single process of the job.
<b>Example</b>	<pre>&gt; msub -l nodes=4:ppn=2,pmem=1024mb</pre> <p><i>The job must have 4 compute nodes with 2 processors per node, and each process of the job is limited to 1024 MB of physical memory.</i></p>
PREF	
<b>Format</b>	[{feature variable}:]<STRING>[:<STRING>]...
	<p><b>i</b> If feature or variable are not specified, then feature is assumed.</p>
<b>Description</b>	<p>Specifies which <a href="#">node features</a> are preferred by the job and should be allocated if available. If preferred node criteria are specified, Moab favors the allocation of matching resources but is not bound to only consider these resources.</p> <p><b>i</b> Preferences are not honored unless the <a href="#">node allocation</a> policy is set to <a href="#">PRIORITY</a> and the <a href="#">PREF</a> priority component is set within the node's <a href="#">PRIORITYF</a> attribute.</p>
<b>Example</b>	<pre>&gt; qsub -l nodes=1,pref=bigmem</pre> <p><i>The job may run on any nodes but prefers to allocate nodes with the <b>bigmem</b> feature.</i></p>
PROCS	
<b>Format</b>	<INTEGER>

PROCS	
<b>Description</b>	<p>Requests a specific amount of processors for the job. Instead of users trying to determine the amount of nodes they need, they can instead decide how many processors they need and Moab will automatically request the appropriate amount of nodes from the RM. This also works with feature requests, such as <code>procs=12[:feature1[:feature2[-]]]</code>.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> Using this resource request overrides any other processor or node related request, such as <code>nodes=4</code>.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 5px;">&gt; msub -l procs=32 myjob.pl</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"><i>Moab will request as many nodes as is necessary to meet the 32-processor requirement for the job.</i></p>

PROLOGUE	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies a user owned prologue script which will be run after the system prologue and <code>prologue.user</code> scripts at the beginning of a job. The syntax is <code>prologue=&lt;file&gt;</code>. The file can be designated with an absolute or relative path.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> This parameter works only with TORQUE.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">&gt; msub -l prologue=prologue_script.sh job.s</pre>

PVMEM	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specify the maximum amount of virtual memory used by any single process in the job.
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 5px;">&gt; msub -l nodes=4:ppn=2,pvmem=1024mb</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"><i>The job must have 4 compute nodes with 2 processors per node, and each process of the job is limited to 1024 MB of virtual memory.</i></p>

QoS	
<b>Format</b>	<STRING>
<b>Description</b>	Requests the specified QoS for the job.
<b>Example</b>	<pre>&gt; qsub -l walltime=1000,qos=highprio biojob.cmd</pre>

QUEUEJOB	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Indicates whether or not the scheduler should queue the job if resources are not available to run the job immediately
<b>Example</b>	<pre>&gt; msub -l nodes=1,queuejob=false test.cmd</pre>

REQATTR	
<b>Format</b>	Required node attributes with version number support: <i>reqattr</i> =[<must must not should should not>]:<ATTRIBUTE>[> = < =] <VERSION>
<b>Description</b>	<p>Indicates required node attributes. Values may include letters, numbers, dashes, underscores, and spaces.</p> <p>You can choose one of four requirement types for each node attribute you request:</p> <ul style="list-style-type: none"> <li>• <b>must</b> – The node on which this job runs must include the attribute at the value specified. If no node matches this requirement, Moab will not schedule the job.</li> <li>• <b>must not</b> – The node on which this job runs must not include the attribute at the value specified. If no node matches this requirement, Moab will not schedule the job.</li> <li>• <b>should</b> – If possible, the node on which this job runs should include the attribute at the value specified. If no node matches this requirement, Moab selects a node without it.</li> <li>• <b>should not</b> – If possible, the node on which this job runs should not include the attribute at the value specified. If no node matches this requirement, Moab selects a node without it.</li> </ul> <p>If you do not specify a requirement type, Moab assumes "must."</p> <p>For information about using reqattr to request dynamic features, see <a href="#">Configuring dynamic features in TORQUE and Moab on page 647</a>.</p>
<b>Example</b>	<pre>&gt; qsub -l reqattr=matlab=7.1 testj.txt</pre>

RESFAILPOLICY	
<b>Format</b>	One of <i>CANCEL</i> , <i>HOLD</i> , <i>IGNORE</i> , <i>NOTIFY</i> , or <i>REQUEUE</i>
<b>Description</b>	Specifies the action to take on an executing job if one or more allocated nodes fail. This setting overrides the global value specified with the <a href="#">NODEALLOCRESFAILUREPOLICY</a> parameter.
<b>Example</b>	<pre>&gt; msub -l resfailpolicy=ignore</pre> <p><i>For this particular job, ignore node failures.</i></p>

RMTYPE	
<b>Format</b>	<STRING>
<b>Description</b>	One of the resource manager types currently available within the cluster or grid. Typically, this is one of <i>PBS</i> , <i>LSF</i> , <i>LL</i> , <i>SGE</i> , <i>SLURM</i> , <i>BProc</i> , and so forth.
<b>Example</b>	<pre>&gt; msub -l rmttype=ll</pre> <p>Only run job on a Loadleveler destination resource manager.</p>

SIGNAL	
<b>Format</b>	<INTEGER>[@<OFFSET>]
<b>Description</b>	Specifies the pre-termination signal to be sent to a job prior to it reaching its walltime limit or being terminated by Moab. The optional offset value specifies how long before job termination the signal should be sent. By default, the pre-termination signal is sent one minute before a job is terminated
<b>Example</b>	<pre>&gt; msub -l signal=32@120 bio45.cmd</pre>

SPRIORITY	
<b>Format</b>	<INTEGER>
<b>Default</b>	0

SPRIORITY	
<b>Description</b>	Allows Moab administrators to set a system priority on a job (similar to <a href="#">setspri</a> ). This only works if the job submitter is an administrator.
<b>Example</b>	<pre>&gt; qsub -l nodes=16,spriority=100 job.cmd</pre>

TEMPLATE	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies a job template to be used as a <a href="#">set</a> template. The requested template must have SELECT-T=TRUE (See <a href="#">Job Templates</a> ).
<b>Example</b>	<pre>&gt; msub -l walltime=1000,nodes=16,template=biojob job.cmd</pre>

TERMTIME	
<b>Format</b>	<TIMESPEC>
<b>Default</b>	0
<b>Description</b>	Specifies the time at which Moab should cancel a queued or active job (See <a href="#">Job Deadline Support</a> ).
<b>Example</b>	<pre>&gt; msub -l nodes=10,walltime=600,termtime=12:00_Jun/14 job.cmd</pre>

TPN	
<b>Format</b>	<INTEGER>[+]
<b>Default</b>	0

TPN	
<b>Description</b>	<p>Tasks per node allowed on allocated hosts. If the plus (+) character is specified, the tasks per node value is interpreted as a minimum tasks per node constraint; otherwise it is interpreted as an exact tasks per node constraint.</p> <p><b>Differences between TPN and PPN:</b></p> <p>There are two key differences between the following: (A) <code>qsub -l nodes=12:ppn=3</code> and (B) <code>qsub -l nodes=12,tpn=3</code>.</p> <p>The first difference is that <b>ppn</b> is interpreted as the <i>minimum</i> required tasks per node while <b>tpn</b> defaults to exact tasks per node; case (B) executes the job with exactly 3 tasks on each allocated node while case (A) executes the job with at least 3 tasks on each allocated node-  <code>nodeA: 4, nodeB: 3, nodeC: 5</code></p> <p>The second major difference is that the line, <code>nodes=X:ppn=Y</code> actually requests <math>X*Y</math> tasks, whereas <code>nodes=X, tpn=Y</code> requests only X tasks.</p> <p><b>TPN with TORQUE as an RM:</b></p> <p>Moab interprets nodes loosely as procs. TORQUE interprets nodes as the number of nodes from the actual number of nodes that you have in your nodes file, not your total number of procs. This means that if TORQUE is your resource manager and you specify <code>msub -l nodes=16:tpn=8</code> but do not have 16 nodes, TORQUE will not run the job. Instead, you should specify <code>msub -l procs=16:tpn=8</code>.</p> <p>To resolve the problem long term, you can also set <code>server resources_available.nodect</code> to the total number of procs in your system and use <code>msub -l nodes=16:tpn=8</code> as you would in a non-TORQUE Moab environment. For more information, see <a href="#">resources available on page 2634</a> in the TORQUE Administrator Guide.</p>
<b>Example</b>	<pre>&gt; msub -l nodes=10,walltime=600,tpn=4 job.cmd</pre>

TRIG	
<b>Format:</b>	<TRIGSPEC>
<b>Description:</b>	<p>Adds trigger(s) to the job (See <a href="#">Creating a trigger on page 748</a> for specific syntax.).</p> <div style="border: 1px solid #0070c0; padding: 5px; background-color: #e6f2ff;"> <p> Job triggers can only be specified if allowed by the QoS flag <code>trigger</code>. See <a href="#">Enabling job triggers on page 755</a> for more information.</p> </div>
<b>Example:</b>	<pre>&gt; qsub -l trig=etype=start\&amp;atype=exec\&amp;action="/tmp/email.sh job.cmd"</pre>

TRL (Format 1)	
<b>Format</b>	<INTEGER>[@<INTEGER>] [:<INTEGER>[@<INTEGER>]] . . .

TRL (Format 1)	
<b>Default:</b>	0
<b>Description:</b>	Specifies alternate task requests with their optional walltimes (See <a href="#">Malleable Jobs</a> ).
<b>Example:</b>	<pre>&gt; msub -l trl=2@500:4@250:8@125:16@62 job.cmd</pre> <p>or</p> <pre>&gt; qsub -l trl=2:3:4</pre>

TRL (Format 2)	
<b>Format</b>	<INTEGER>-<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies a range of task requests that require the same walltime (See <a href="#">Malleable Jobs</a> ).
<b>Example</b>	<pre>&gt; msub -l trl=32-64 job.cmd</pre> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> For optimization purposes Moab does not perform an exhaustive search of all possible values but will at least do the beginning, the end, and 4 equally distributed choices in between.</p> </div>

VAR	
<b>Format</b>	<ATTR>[:<VALUE>]
<b>Description</b>	Adds a generic variable or variables to the job.
<b>Example</b>	<pre>&gt; msub -l VAR=testvar1:testvalue1</pre> <p style="text-align: center;"><i>Single variable</i></p> <pre>&gt; msub -l VAR=testvar1:testvalue1+testvar2:testvalue2+testvar3:testvalue3</pre> <p style="text-align: center;"><i>Multiple variables</i></p>

VC	
<b>Format</b>	vc=<NAME>
<b>Description</b>	Submits the job or workflow to a virtual container (VC).
<b>Example</b>	<code>vc=vc13</code>

VMEM	
<b>Format:</b>	<INTEGER>
<b>Description:</b>	Specify the maximum amount of virtual memory used by all concurrent processes in the job.
<b>Example:</b>	<pre>&gt; msub -l nodes=4:ppn=2,vmem=1024mb</pre> <p><i>The job must have 4 compute nodes with 2 processors per node, and the job is limited to 1024 MB of virtual memory.</i></p>

## Resource Manager Extension Examples

If more than one extension is required in a given job, extensions can be concatenated with a semicolon separator using the format <ATTR>:<VALUE>[;<ATTR>:<VALUE>]...

Example 3-144:

```
#@comment="HOSTLIST:node1,node2;QoS:special;SID:silverA"
```

*Job must run on nodes node1 and node2 using the QoS special. The job is also associated with the system ID silverA allowing the silver daemon to monitor and control the job.*

Example 3-145:

```
# PBS -W x="\NODESET:ONEOF:NETWORK;DMEM:64\""
```

*Job will have resources allocated subject to network based nodeset constraints. Further, each task will dedicate 64 MB of memory.*

Example 3-146:

```
> qsub -l nodes=4,walltime=1:00:00 -W x="FLAGS:ADVRES:john.1"
```

*Job will be forced to run within the john.1 reservation.*

## Configuring dynamic features in TORQUE and Moab

Used together, the [reqattr](#) RM extension and TORQUE [\\$varattr on page 2799](#) parameter allow you to create jobs that request resources that may change or disappear. For example, if you wanted a job to request a certain version of Octave but different versions are configured on each node and updated at any time, you can create a script that searches for the feature and version on the nodes at a specified interval. Your Moab job can then retrieve the dynamic node attributes from the latest poll and use them for scheduling.

This functionality is available when you use the TORQUE `$varattr` parameter to configure a script that regularly retrieves updates on the nodes' feature(s) and the `reqattr` RM extension to require a feature with a certain value.

### To set up a dynamic feature in TORQUE and Moab

1. Create a script that pulls the information you need. For instance, the following script pulls the version of Octave on each node and prints it.

```
#!/bin/bash
# pull the version string for octave and print it for $varattr
version_str=`octave -v | grep version`
[[ $version_str =~ ([:digit:]).[:digit:].[:digit:] ]]
echo "Octavē: ${BASH_REMATCH[1]}"
```

2. Use the TORQUE `$varattr` parameter to configure the script. Specify both the number of seconds between each time TORQUE runs the script and the path to the script. If you set the seconds to `-1`, the script will run just once. You may include arguments if desired. In the following example, the `varattr` parameter specifies that TORQUE calls the Octave script every 30 seconds.

```
$varattr 30 /usr/local/scripts/octave.sh
```

3. Submit your job in Moab, specifying `reqattr` as a resource. In this example, the job requests a node where the octave feature has a value of 3.2.4 (that the node has Octave version 3.2.4 installed).

```
> msub -l rerqattr=octave=3.2.4 myJob.sh
```

*Your job requests a node with Octave version 3.2.4. TORQUE passes the most recent (pulled within the last 30 seconds) version of Octave on each node. Moab then schedules the job on a node that currently has Octave 3.2.4.*

### Related topics

- [Resource Manager Overview](#)

## PBS Resource Manager Extensions

Resource manager extensions within PBS are used by setting the `-W` flag. To enable this flag, some versions of PBS must be rebuilt. [TORQUE](#) and recent OSCAR distributions come with the flag enabled by default. Most other versions do not. The required steps are documented in what follows:

- 1.

```
> qterm -t quick
#shutdown PBS server
```

cd to the directory from which you executed the PBS 'configure' at install time

```
> make distclean  
> ./configure <WITH OPTIONS>
```

2. Create [addparam](#) script  
(`chmod +x addparam`)

3. 

```
> addparam x  
> make
```

Backup current `$PBS_HOMEDIR` directory contents

 `$PBS_HOMEDIR` defaults to `/usr/spool/PBS`.

4. 

```
> make install
```

Restore old `$PBS_HOMEDIR` directory contents

5. 

```
> pbs_server # restart PBS server
```

A job's QOS level can then be specified using the `qsub -W` flag. For example, `qsub -W x=iQOS:hi -l nodes=4 ...`

```

#!/bin/sh
#script:  addparam
#usage:  addparam $Parameter [S|L]
NewParameter=$1
ParameterType=x$2
if [ ! -d src/include ]; then
    echo "error: `basename $0` src/include doesn't exist, run configure"
    1>&2
    exit 1
fi
# run make in this directory to pull over the template files
cd src/include
if make
then
    if grep -q "\"$NewParameter\"" site_*.h 2>/dev/null; then
        echo "parameter $NewParameter previously added"
        exit 0
    fi
fi
chmod +w site_job_attr_enum.h
echo "
    JOB_SITE_ATR_$1,
" >> site_job_attr_enum.h
chmod +w site_job_attr_def.h
if [ $ParameterType = "xS" ]
then
    echo "
        {   \"$NewParameter\",
            decode_str,
            encode_str,
            set_str,
            comp_str,
            free_str,
            NULL_FUNC,
            READ_WRITE,
            ATR_TYPE_STR,
            PARENT_TYPE_JOB
        },
" >> site_job_attr_def.h
else
    echo "
        {   \"$NewParameter\",
            decode_l,
            encode_l,
            set_l,
            comp_l,
            free_null,
            NULL_FUNC,
            READ_WRITE,
            ATR_TYPE_LONG,
            PARENT_TYPE_JOB
        },
" >> site_job_attr_def.h
fi
exit 0

```

## Adding New Resource Manager Interfaces

Moab interfaces with numerous resource management systems. Some of these interact through a resource manager specific interface (OpenPBS/PBSPro, Loadleveler, LSF), while others interact through

generalized interfaces such as SSS or Wiki (See the [Wiki Overview](#)). For most resource managers, either route is possible depending on where it is easiest to focus development effort. Use of Wiki generally requires modifications on the resource manager side while creation of a new resource manager specific Moab interface would require more changes to Moab modules.

Regardless of the interface approach selected, adding support for a new resource manager is typically a straightforward process for about 95% of all supported features. The final 5% of features usually requires a bit more effort as each resource manager has a number of distinct concepts that must be addressed.

- [Resource Manager Specific Interfaces](#)
- [Wiki Interface](#)
- [SSS Interface](#)

## Resource Manager Specific Interfaces

If you require tighter integration and need additional instruction, see [Managing Resources Directly with the Native Interface](#). If you would like consultation on support for a new resource manager type, please [contact](#) the Professional Services group at Adaptive Computing.

## Wiki Interface

The Wiki interface is already defined as a resource manager type, so no modifications are required within Moab. Additionally, no resource manager specific library or header file is required. However, within the resource manager, internal job and node objects and attributes must be manipulated and placed within Wiki based interface concepts as defined in the [Wiki Overview](#). Additionally, resource manager parameters must be created to allow a site to configure this interface appropriately.

## SSS Interface

The SSS interface is an XML based generalized resource manager interface. It provides an extensible, scalable, and secure method of querying and modifying general workload and resource information.

### Related topics

- [Creating New Tools within the Native Resource Manager Interface](#)

## Managing Resources Directly with the Native Interface

- [Native Interface Overview](#)
- [Configuring the Native Interface](#)
  - [Configuring the Resource Manager](#)
  - [Reporting Resources](#)

- [Generating Cluster Query Data](#)
  - [Flat Cluster Query Data](#)
  - [Interfacing to FLEXlm](#)
  - [Interfacing to Nagios](#)
  - [Interfacing to Supermon](#)
- [Configuring Resource Types](#)
- [Creating New Tools to Manage the Cluster](#)

## Native Interface Overview

The Native interface allows a site to augment or even fully replace a resource manager for managing resources. In some situations, the full capabilities of the resource manager are not needed and a lower cost or lower overhead alternative is preferred. In other cases, the nature of the environment may make use of a resource manager impossible due to lack of support. Still, in other situations it is desirable to provide information about additional resource attributes, constraints, or state from alternate sources.

In any case, Moab provides the ability to directly query and manage resources along side of or without the use of a resource manager. This interface, called the NATIVE interface can also be used to launch, cancel, and otherwise manage jobs. This NATIVE interface offers several advantages including the following:

- No cost associated with purchasing a resource manager
- No effort required to install or configure the resource manager
- Ability to support abstract resources
- Ability to support abstract jobs
- Ability to integrate node availability information from multiple sources
- Ability to augment node configuration and utilization information provided by a resource manager

However, the NATIVE interface may also have some drawbacks.

- No support for standard job submission languages
- Limited default configured and utilized resource tracking (additional resource tracking available with additional effort)

At a high level, the native interface works by launching threaded calls to perform standard resource manager activities such as managing resources and jobs. The desired calls are configured within Moab and used whenever an action or updated information is required.

## Configuring the Native Interface

Using the native interface consists of defining the interface type and location. As mentioned earlier, a single object may be fully defined by multiple interfaces simultaneously with each interface updating a particular aspect of the object.

## Configuring the Resource Manager

The Native resource manager must be configured using the [RMCFG](#) parameter. To specify the native interface, the **TYPE** attribute must be set to *NATIVE*.

```
RMCFG[local] TYPE=NATIVE
RMCFG[local] CLUSTERQUERYURL=exec:///tmp/query.sh
```

## Reporting Resources

To indicate the source of the resource information, the **CLUSTERQUERYURL** attribute of the [RMCFG](#) parameter should be specified. This attribute is specified as a URL where the protocols *FILE*, *EXEC* and *SQL* are allowed. If a protocol is not specified, the protocol *EXEC* is assumed.

Format	Description
<b>EXEC</b>	Execute the script specified by the URL path. Use the script stdout as data.
<b>FILE</b>	Load the file specified by the URL path. Use the file contents as data.
<b>SQL</b>	Load data directly from an SQL database using the FULL format described below.

Moab considers a NativeRM script to have failed if it returns with a non-zero exit code or if the [CHILDSTDERRCHECK](#) parameter is set and its appropriate conditions are met. In addition, the NativeRM script associated with a job submit URL will be considered as having failed if its standard output stream contains the text `ERROR`.

This simple example queries a file on the server for information about every node in the cluster. This differs from Moab remotely querying the status of each node individually.

```
RMCFG[local] TYPE=NATIVE
RMCFG[local] CLUSTERQUERYURL=file:///tmp/query.txt
```

## Generating Cluster Query Data

### Flat Cluster Query Data

If the *EXEC* or *FILE* protocol is specified in the **CLUSTERQUERYURL** attribute, the data should provide flat text strings indicating the state and attributes of the node. The format follows the [Moab Resource Manager Language Interface Specification](#) where attributes are delimited by white space rather than ';' (See [Resource Data Format](#)):

Describes any set of node attributes with format: `<NAME><ATTR>=<VAL> [<ATTR>=<VAL>] . . .`

<b>&lt;NAME&gt;</b>	Name of node
<b>&lt;ATTR&gt;</b>	Node attribute

<b>&lt;VAL&gt;</b>	Value of node attribute
--------------------	-------------------------

```
n17 CPROC=4 AMEMORY=100980 STATE=idle
```

## Interfacing to FLEXlm

Moab can interface with FLEXlm to provide scheduling based on [license](#) availability. Informing Moab of license dependencies can reduce the number of costly licenses required by your cluster by allowing Moab to intelligently schedule around license limitations.

Provided with Moab in the tools directory is a Perl script, `license.mon.flexLM.pl`. This script queries a FLEXlm license server and gathers data about available licenses. This script then formats this data for Moab to read through a native interface. This script can easily be used by any site to help facilitate FLEXlm integration--the only modification necessary to the script is setting the `@FLEXlmCmd` to specify the local command to query FLEXlm. To make this change, edit `license.mon.flexLM.pl` and, near the top of the file, look for the line:

```
my @FLEXlmCmd = ("SETME");
```

Set the `@FLEXlmCmd` to the appropriate value for your system to query a license server and license file (if applicable). If `lmutil` is not in the `PATH` variable, specify its full path. Using the `lmutil -a` argument will cause it to report all licenses. The `-c` option can be used to specify an optional license file.

To test this script, run it manually. If working correctly, it will produce output similar to the following:

```
> ./license.mon.flexLM.pl
GLOBAL UPDATETIME=1104688300 STATE=idle ARES=autoCAD:130,id1_mpeg:160
CRES=autoCAD:200,id1_mpeg:330
```

If the output looks incorrect, set the `$LOGLEVEL` variable inside of `license.mon.flexLM.pl`, run it again, and address the reported failure.

Once the license interface script is properly configured, the next step is to add a *license* native resource manager to Moab via the `moab.cfg` file:

```
RMCFG[FLEXlm] TYPE=NATIVE RESOURCETYPE=LICENSE
RMCFG[FLEXlm] CLUSTERQUERYURL=exec://$TOOLS_DIR/flexlm/license.mon.flexLM.pl
...
```

Once this change is made, restart Moab. The command `mdiag -R` can be used to verify that the resource manager is properly configured and is in the state `Active`. Detailed information regarding configured and utilized licenses can be viewed by issuing the `mdiag -n`. Floating licenses (non-node-locked) will be reported as belonging to the `GLOBAL` node.

**i** Due to the inherent conflict with the plus sign (+), the provided license manager script replaces occurrences of the plus sign in license names with the underscore symbol (`_`). This replacement requires that licenses with a plus sign in their names be requested with an underscore in place of any plus signs.

## Interfacing to Multiple License Managers Simultaneously

If multiple license managers are used within a cluster, Moab can interface to each of them to obtain the needed license information. In the case of FLEXlm, this can be done by making one copy of the `license.mon.flexLM.pl` script for each license manager and configuring each copy to point to a different license manager. Then, within Moab, create one native resource manager interface for each license manager and point it to the corresponding script as in the following example:

```

RMCFG [FLEXlm1]    TYPE=NATIVE RESOURCETYPE=LICENSE
RMCFG [FLEXlm1]    CLUSTERQUERYURL=exec://$TOOLSDIR/flexlm/license.mon.flexLM1.pl
RMCFG [FLEXlm2]    TYPE=NATIVE RESOURCETYPE=LICENSE
RMCFG [FLEXlm2]    CLUSTERQUERYURL=exec://$TOOLSDIR/flexlm/license.mon.flexLM2.pl
RMCFG [FLEXlm3]    TYPE=NATIVE RESOURCETYPE=LICENSE
RMCFG [FLEXlm3]    CLUSTERQUERYURL=exec://$TOOLSDIR/flexlm/license.mon.flexLM3.pl
...

```

**i** For an overview of license management, including job submission syntax, see [License Management](#).

**i** It may be necessary to increase the default limit, `MMAx_GRES`. See [Appendix D](#) for more implementation details.

## Interfacing to Nagios

Moab can interface with Nagios to provide scheduling based on network hosts and services availability.

Nagios installation and configuration documentation can be found at [Nagios.org](http://Nagios.org).

Provided with Moab in the tools directory is a Perl script, `node.query.nagios.pl`. This script reads the Nagios `status.dat` file and gathers data about network hosts and services. This script then formats data for Moab to read through a native interface. This script can be used by any site to help facilitate Nagios integration. To customize the data that will be formatted for Moab, make the changes in this script.

You may need to customize the associated configuration file in the `etc` directory, `config.nagios.pl`. The `statusFile` line in this script tells Moab where the Nagios `status.dat` file is located. Make sure that the path name specified is correct for your site. Note that the interval which Nagios updates the Nagios `status.dat` file is specified in the Nagios `nagios.cfg` file. Refer to Nagios documentation for further details.

To make these changes, familiarize yourself with the format of the Nagios `status.dat` file and make the appropriate additions to the script to include the desired Moab RM language (formerly WIKI) Interface attributes in the Moab output.

To test this script, run it manually. If working correctly, it will produce output similar to the following:

```

> ./node.query.nagios.pl
gateway STATE=Running
localhost STATE=Running CPULOAD=1.22 ADISK=75332

```

Once the Nagios interface script is properly configured, the next step is to add a Nagios native resource manager to Moab via the `moab.cfg` file:

```

RMCFG[nagios] TYPE=NATIVE
RMCFG[nagios] CLUSTERQUERYURL=exec://$TOOLS_DIR/node.query.nagios.pl
...

```

Once this change is made, restart Moab. The command `mdiag -R` can be used to verify that the resource manager is properly configured and is in the state `Active`. Detailed information regarding configured Nagios node information can be viewed by issuing the `mdiag -n -v`.

```

> mdiag -n -v
compute node summary
Name           State  Procs      Memory      Disk      Swap
Speed  Opsys  Arch Par   Load Rsv Classes
Features
gateway        Running  0:0      0:0      0:0      0:0
1.00          -      - dav  0.00  0 -
-
WARNING: node 'gateway' is busy/running but not assigned to an active job
WARNING: node 'gateway' has no configured processors
localhost      Running  0:0      0:0      75343:75347  0:0
1.00          -      - dav  0.48  0 -
-
WARNING: node 'localhost' is busy/running but not assigned to an active job
WARNING: node 'localhost' has no configured processors
-----
Total Nodes: 2 (Active: 2 Idle: 0 Down: 0)

```

## Interfacing to Supermon

Moab can integrate with Supermon to gather additional information regarding the nodes in a cluster. A Perl script is provided in the tools directory that allows Moab to connect to the Supermon server. By default the Perl script assumes that Supermon has been started on port 2709 on localhost. These defaults can be modified by editing the respective parameter in `config.supermon.pl` in the `etc` directory. An example setup is shown below.

```

RMCFG[TORQUE] TYPE=pbs
RMCFG[supermon] TYPE=NATIVE CLUSTERQUERYURL=exec://$HOME/tools/node.query.supermon.pl

```

To confirm that Supermon is properly connected to Moab, issue `mdiag -R -v`. The output should be similar to the following example, specifically there are no errors about the `CLUSTERQUERYURL`.

```

diagnosing resource managers
RM[TORQUE] State: Active
  Type:          PBS ResourceType: COMPUTE
  Server:        keche
  Version:       '2.2.0-snap.200707181818'
  Job Submit URL: exec:///usr/local/bin/qsub
  Objects Reported: Nodes=3 (6 procs) Jobs=0
  Flags:        executionServer
  Partition:    TORQUE
  Event Management: EPORT=15004 (no events received)
  Note: SSS protocol enabled
  Submit Command: /usr/local/bin/qsub
  DefaultClass: batch
  RM Performance: AvgTime=0.26s MaxTime=1.04s (4 samples)
  RM Languages:  PBS
  RM Sub-Languages: -
RM[supermon] State: Active
  Type:          NATIVE ResourceType: COMPUTE
  Cluster Query URL: exec://$HOME/node.query.supermon.pl
  Objects Reported: Nodes=3 (0 procs) Jobs=0
  Partition:    supermon
  Event Management: (event interface disabled)
  RM Performance: AvgTime=0.03s MaxTime=0.11s (4 samples)
  RM Languages:  NATIVE
  RM Sub-Languages: -

Note: use 'mrmctl -f messages ' to clear stats/failures

```

Run the Perl script by itself. The script's results should look similar to this:

```

vm01 GMETRIC[CPULOAD]=0.571428571428571 GMETRIC[NETIN]=133 GMETRIC[NETOUT]=702 GMETRIC
[NETUSAGE]=835
vm02 GMETRIC[CPULOAD]=0.428571428571429 GMETRIC[NETIN]=133 GMETRIC[NETOUT]=687 GMETRIC
[NETUSAGE]=820
keche GMETRIC[CPULOAD]=31 GMETRIC[NETIN]=5353 GMETRIC[NETOUT]=4937 GMETRIC[NETUSAGE]
=10290

```

If the preceding functioned properly, issue a `checknode` command on one of the nodes that Supermon is gathering statistics for. The output should look similar to below.

```

node keche
State:      Idle (in current state for 00:32:43)
Configured Resources: PROCS: 2 MEM: 1003M SWAP: 3353M DISK: 1M
Utilized Resources: ---
Dedicated Resources: ---
Generic Metrics: CPULOAD=33.38,NETIN=11749.00,NETOUT=9507.00,NETUSAGE=21256.00
MTBF(longterm): INFINITY MTBF(24h): INFINITY
Opsys:      linux Arch:      ---
Speed:      1.00 CPUload:    0.500
Network Load: 0.87 kB/s
Flags:      rmdetected
Network:    DEFAULT
Classes:    [batch 2:2][interactive 2:2]
RM[TORQUE]: TYPE=PBS
EffNodeAccessPolicy: SHARED
Total Time: 2:03:27 Up: 2:03:27 (100.00%) Active: 00:00:00 (0.00%)
Reservations: ---

```

## Configuring Resource Types

Native Resource managers can also perform special tasks when they are given a specific resource type. These types are specified using the [RESOURCE TYPE](#) attribute of the [RMCFG](#) parameter.

Type	Description
<b>COMPUTE</b>	Normal compute resources (no special handling)
<b>FS</b>	File system resource manager (see <a href="#">Multiple Resource Managers</a> for an example)
<b>LICENSE</b>	Software license manager (see <a href="#">Interfacing with FLEXlm</a> and <a href="#">License Management</a> )
<b>NETWORK</b>	Network resource manager
<b>PROV</b>	Provisioning resource manager. This is the RM Moab uses to modify the OS of a node (not a VM) and to power a node on or off.

## Creating New Tools to Manage the Cluster

Using the scripts found in the `$TOOLS DIR ($INSTDIR/tools)` directory as a template, new tools can be quickly created to monitor or manage most any resource. Each tool should be associated with a particular resource manager service and specified using one of the following resource manager URL attributes.

CLUSTERQUERYURL	
<b>Description</b>	Queries resource state, configuration, and utilization information for compute nodes, networks, storage systems, software licenses, and other resources. For more details, see <a href="#">RM configuration</a> .
<b>Output</b>	Node status and configuration for one or more nodes. See <a href="#">Resource Data Format</a> .
<b>Example</b>	<pre>RMCFG[v-stor] CLUSTERQUERYURL=exec://\$HOME/storquery.pl</pre> <p><i>Moab will execute the storquery.pl script to obtain information about 'v-stor' resources.</i></p>

JOBMODIFYURL	
<b>Description</b>	Modifies a job or application. For more details, see <a href="#">RM configuration</a> .

JOBMODIFYURL	
<b>Input</b>	<code>[-j &lt;JOBEXPR&gt;] [--s[et] --c[lear] --i[ncrement] --d[ecrement]] &lt;ATTR&gt; [=&lt;VALUE&gt;] [&lt;ATTR&gt;[=&lt;VALUE&gt;]]...</code>
<b>Example</b>	<pre>RMCFG[v-stor] JOBMODIFYURL=exec://\$HOME/jobmodify.pl</pre> <p><i>Moab will execute the jobmodify.pl script to modify the specified job.</i></p>

JOBREQUEUEURL	
<b>Description</b>	Requeues a job.
<b>Input</b>	<code>&lt;JOBID&gt;</code>
<b>Example</b>	<pre>RMCFG[v-stor] JOBREQUEUEURL=exec://\$HOME/requeue.pl</pre> <p><i>Moab will execute the requeue.pl script to requeue jobs.</i></p>

JOBRESUMEURL	
<b>Description</b>	Resumes a suspended job or application.
<b>Input</b>	<code>&lt;JOBID&gt;</code>
<b>Example</b>	<pre>RMCFG[v-stor] JOBRESUMEURL=exec://\$HOME/jobresume.pl</pre> <p><i>Moab will execute the jobresume.pl script to resume suspended jobs.</i></p>

JOBSTARTURL	
<b>Description</b>	Launches a job or application on a specified set of resources.

JOBSTARTURL	
<b>Input</b>	<JOBID><TASKLIST><USERNAME> [ARCH=<ARCH>] [OS=<OPSYS>] [IDATA=<STAGEINFILEPATH>[, <STAGEINFILEPATH>]...] [EXEC=<EXECUTABLEPATH>]
<b>Example</b>	<pre>RMCFG[v-stor] JOBSTARTURL=exec://\$HOME/jobstart.pl</pre> <p><i>Moab will execute the jobstart.pl script to execute jobs.</i></p>

JOBSUBMITURL	
<b>Description</b>	Submits a job to the resource manager, but it does not execute the job. The job executes when the <b>JOBSTARTURL</b> is called.
<b>Input</b>	[ACCOUNT=<ACCOUNT>] [ERROR=<ERROR>] [GATTR=<GATTR>] [GNAME=<GNAME>] [GRES=<GRES>:<Value>[, <GRES>:<Value>]*] [HOSTLIST=<HOSTLIST>] [INPUT=<INPUT>] [IWD=<IWD>] [NAME=<NAME>] [OUTPUT=<OUTPUT>] [RCLASS=<RCLASS>] [REQUEST=<REQUEST>] [RFEATURES=<RFEATURES>] [RMFLAGS=<RMFLAGS>] [SHELL=<SHELL>] [TASKLIST=<TASKLIST>] [TASKS=<TASKS>] [TEMPLATE=<TEMPLATE>] [UNAME=<UNAME>] [VARIABLE=<VARIABLE>] [WCLIMIT=<WCLIMIT>] [ARGS=<Value>[ <Value>]*]
	<div style="border: 1px solid black; padding: 5px;"> <p><b>i</b> ARGS must be the last submitted attribute because there can be multiple space-separated values for ARGS.</p> </div>
<b>Example</b>	<pre>RMCFG[v-stor] JOBSUBMITURL=exec://\$HOME/jobsubmit.pl</pre> <p><i>Moab submits the job to the jobsubmit.pl script for future job execution.</i></p>

JOBSUSPENDURL	
<b>Description</b>	Suspends in memory an active job or application.
<b>Input</b>	<JOBID>
<b>Example</b>	<pre>RMCFG[v-stor] JOBSUSPENDURL=exec://\$HOME/jobsuspend.pl</pre> <p><i>Moab will execute the jobsuspend.pl script to suspend active jobs.</i></p>

NODEMODIFYURL	
<b>Description</b>	Provide method to dynamically modify/provision compute resources including operating system, applications, queues, node features, power states, etc.
<b>Input</b>	<NODEID>[, <NODEID>] [--force] [--set <ATTR>=<VAL> --clear <ATTR>} ATTR is one of the node attributes listed in <a href="#">Resource Data Format</a>
<b>Example</b>	<pre>RMCFG[warewulf] NODEMODIFYURL=exec://\$HOME/provision.pl</pre> <p><i>Moab will reprovision compute nodes using the provision.plscript.</i></p>

NODEPOWERURL	
<b>Description</b>	Allows Moab to issue IPMI power commands.
<b>Input</b>	<NODEID>[, <NODEID>] ON   OFF
<b>Example</b>	<pre>RMCFG[node17rm] NODEPOWERURL=exec://\$TOOLS DIR/ipmi.power.pl</pre> <p><i>Moab will issue a power command contained in the ipmi.power.plscript.</i></p>

SYSTEMMODIFYURL	
<b>Description</b>	Provide method to dynamically modify aspects of the compute environment which are directly associated with cluster resources. For more details, see <a href="#">RM configuration</a> .

SYSTEMQUERYURL	
<b>Description</b>	Provide method to dynamically query aspects of the compute environment which are directly associated with cluster resources. For more details, see <a href="#">RM configuration</a> .
<b>Input</b>	default <ATTR> ATTR is one of images
<b>Output</b>	<STRING>

## SYSTEMQUERYURL

### Example

```
RMCFG[warewulf] SYSTEMQUERYURL=exec://$HOME/checkimage.pl
```

*Moab will load the list of images available from warewulf using the `checkimage.pl` script.*

## WORKLOADQUERYURL

### Description:

Provide method to dynamically query the system workload (jobs, services, etc.) of the compute environment which are associated with managed resources.

**i** Job/workload information should be reported back from the URL (script, file, web service, etc.) using the [Moab RM language](#) (formerly WIKI).

For more details, see [RM configuration](#).

### Output:

<STRING>

### Example:

```
RMCFG[xt] WORKLOADQUERYURL=exec://$HOME/job.query.xt3.pl
```

*Moab will load job/workload information by executing the `job.query.xt3.pl` script.*

### Related topics

- [mdiag -R](#) command (evaluate resource managers)
- [License Management](#)
- [Moab Resource Manager Language Data Format](#)
- [Managing Resources with SLURM](#)

## Utilizing Multiple Resource Managers

### Multi-RM Overview

In many instances a site may have certain resources controlled by different resource managers. For example, a site may use a particular resource manager for licensing software for jobs, another resource manager for managing file systems, another resource manager for job control, and another for node monitoring. Moab can be configured to communicate with each of these resource managers, gathering all their data and incorporating such into scheduling decisions. With a more distributed approach to resource handling, failures are more contained and scheduling decisions can be more intelligent.

## Configuring Multiple Independent Resource Manager Partitions

Moab must know how to communicate with each resource manager. In most instances, this is simply done by configuring a [query command](#).

## Migrating Jobs between Resource Managers

With multi-resource manager support, a job may be submitted either to a local resource manager queue or to the Moab global queue. In most cases, submitting a job to a resource manager queue constrains the job to only run within the resources controlled by that resource manager. However, if the job is submitted to the Moab global queue, it can use resources of any active resource manager. This is accomplished through job translation and staging.

When Moab evaluates resource availability, it determines the cost in terms of both data and job staging. If staging a job's executable or input data requires a significant amount of time, Moab integrates data and compute resource availability to determine a job's earliest potential start time on a per resource manager basis and makes an optimal scheduling decision accordingly. If the optimal decision requires a data stage operation, Moab reserves the required compute resources, stages the data, and then starts the job when the required data and compute resources are available.

## Aggregating Information into a Cohesive Node View

Using the native interface, Moab can actually perform most of these functions without the need for an external resource manager. First, configure the native resource managers:

```
RMCFG[base]      TYPE=PBS
RMCFG[network]  TYPE=NATIVE
RMCFG[network]  CLUSTERQUERYURL=/tmp/network.sh
RMCFG[fs]       TYPE=NATIVE
RMCFG[fs]       CLUSTERQUERYURL=/tmp/fs.sh
```

The network script can be as simple as the following:

```
> _RX=`/sbin/ifconfig eth0 | grep "RX by" | cut -d: -f2 | cut -d' ' -f1`; \
> _TX=`/sbin/ifconfig eth0 | grep "TX by" | cut -d: -f3 | cut -d' ' -f1`; \
> echo `hostname` GMETRIC[netusage]=`echo "$_RX + $_TX" | bc`;
```

The preceding script would output something like the following:

```
node01 GMETRIC[netusage]=10928374
```

Moab grabs information from each resource manager and includes its data in the final view of the node.

```

> checknode node01
node node01
State: Running (in current state for 00:00:20)
Configured Resources: PROCS: 2 MEM: 949M SWAP: 2000M disk: 1000000
Utilized Resources: SWAP: 9M
Dedicated Resources: PROCS: 1 disk: 1000
Opsys: Linux-2.6.5-1.358 Arch: linux
Speed: 1.00 CPUload: 0.320
Location: Partition: DEFAULT Rack/Slot: NA
Network Load: 464.11 b/s
Network: DEFAULT
Features: fast
Classes: [batch 1:2][serial 2:2]
Total Time: 00:30:39 Up: 00:30:39 (100.00%) Active: 00:09:57 (32.46%)
Reservations:
  Job '5452'(x1) -00:00:20 -> 00:09:40 (00:10:00)
JobList: 5452

```

Notice that the Network Load is now being reported along with disk usage.

### Example File System Utilization Tracker (per user)

The following configuration can be used to track file system usage on a per user basis:

```

.....
RMCFG[file] TYPE=NATIVE
RMCFG[file] RESOURCETYPE=FS
RMCFG[file] CLUSTERQUERYURL=/tmp/fs.pl
.....

```

Assuming that `/tmp/fs.pl` outputs something of the following [format](#):

```

DEFAULT STATE=idle AFS=<fs id="user1" size="789456"></fs><fs
id="user2" size="123456"></fs>

```

This will track disk usage for users `user1` and `user2` every 24 hours.

## License Management

- [License Management Overview](#)
- [Controlling and Monitoring License Availability](#)
- [Requesting Licenses w/in Jobs](#)

### License Management Overview

Software license management is typically enabled in one of two models: node-locked and floating. Under a node-locked license, use of a given application is constrained to certain hosts. For example, `node013` may support up to two simultaneous jobs accessing application `matlab`. In a floating license model, a limited number of software licenses are made available cluster wide, and these licenses may be used on any combination of compute hosts. In each case, these licenses are consumable and application access is denied once they are gone.

Moab supports both node-locked and floating license models and even allows mixing the two models simultaneously. Moab monitors license usage and only launches an application when required software license availability is guaranteed. In addition, Moab also reserves licenses in conjunction with future jobs to ensure these jobs can run at the appropriate time.

**i** By default, Moab supports up to 128 independent license types.

**i** Moab license recognition is case insensitive. This means that two licenses with the same spelling and different capitalization are still recognized as the same license. When this occurs, Moab considers the license invalid.

## Controlling and Monitoring License Availability

Moab can use one of three methods to determine license availability. These methods include locally specifying [consumable generic resources](#), obtaining consumable generic resource information from the [resource manager](#), and interfacing directly with a [license manager](#).

### Local Consumable Resources

Both node-locked and floating licenses can be locally specified within Moab using the [NODECFG](#) parameter. In all cases, this is accomplished by associating the license with a node using the [GRES](#) (or generic resource) attribute. If floating, the total cluster-wide license count should be associated with the GLOBAL node. If node-locked, the per node license count should be associated with each compute host (or globally using the *DEFAULT* node). For example, if a site has two node-locked licenses for application *EvalA* and six floating licenses for application *EvalB*, the following configuration could be used:

```
NODECFG [node001]  GRES=EvalA:2
NODECFG [node002]  GRES=EvalA:2
NODECFG [GLOBAL]   GRES=EvalB:6
...
```

### Resource Manager Based Consumable Resources

Some resource managers support the ability to define and track generic resource usage at a per node level. In such cases, support for node-locked licenses may be enabled by specifying this information within the resource manager. Moab automatically detects and schedules these resources. For example, in the case of [TORQUE](#), this can be accomplished by adding generic resource specification lines to the [MOM configuration](#) file.

### Interfacing to an External License Manager

Moab may also obtain live software license information from a running license manager. Direct interfaces to supported license managers such as FlexLM may be created using the [Native Resource Manager](#) feature. A complete example on interfacing to an external license manager is provided in the [FLEXlm](#) section of the native resource manager overview.

### Interfacing to Multiple License Managers

Moab may interface to multiple external license managers simultaneously simply by defining additional native resource manager interfaces. See the FLEXlm [Native Resource Manager Overview](#) for more information.

## Requesting Licenses within Jobs

Requesting use of software licenses within jobs is typically done in one of two ways. In most cases, the native resource manager job submission language provides a direct method of license specification; for example, in the case of [TORQUE](#), OpenPBS, or PBSPro, the [software](#) argument could be specified using the format `<SOFTWARE_NAME>[+<LICENSE_COUNT>]` as in the following example:

```
> qsub -l nodes=2,software=blast cmdscript.txt
```

**i** The license count is a job total, not a per task total, and the license count value defaults to 1.

An alternative to direct specification is the use of the Moab [resource manager extensions](#). With these extensions, licenses can be requested as generic resources, using the [GRES](#) attribute. The job in the preceding example could also be requested using the following syntax:

```
> qsub -l nodes=2 -W x=GRES:blast cmdscript.txt
```

In each case, Moab automatically determines if the software licenses are node-locked or floating and applies resource requirements accordingly.

If a job requires multiple software licenses, whether of the same or different types, a user would use the following syntax:

```
> qsub -l nodes=2 -W x=GRES:blast+2 cmdscript.txt # two 'blast' licenses required
> qsub -l nodes=2 -W x=GRES:blast+2%bkeep+3 cmdscript.txt # two 'blast' and three
'bkeep' licenses are required
```

### Related topics

- [Native Resource Manager License Configuration](#)
- License Ownership with [Advance Reservations](#)
- Multi-Cluster License Sharing with [Moab Workload Manager for Grids](#) Interfaces

## Resource Provisioning

- [Resource Provisioning Overview](#)
- [Configuring Provisioning](#)

## Resource Provisioning Overview

When processing a resource request, Moab attempts to match the request to an existing available resource. However, if the scheduler determines that the resource is not available or will not be available

due to load or policy for an appreciable amount of time, it can select a resource to modify to meet the needs of the current requests. This process of modifying resources to meet existing needs is called provisioning.

Currently, there are two types of provisioning supported: operating system (OS) and application. As its name suggests, OS provisioning allows the scheduler to modify the operating system of an existing compute node while application level provisioning allows the scheduler to request that a software application be made available on a given compute node. In each case, Moab evaluates the costs of making the change in terms of time and other resources consumed before making the decision. Only if the benefits outweigh the costs will the scheduler initiate the change required to support the current workload.



Preemption (requeueing) does not work with dynamic provisioning.

## Configuring Provisioning

Enabling provisioning consists of configuring an interface to a provisioning manager, specifying which nodes can take advantage of this service, and what the estimated cost and duration of each change will be. This interface can be used to contact provisioning software such as [xCat](#) or HP's Server Automation tool. Additionally, locally developed systems can be interfaced via a script or web service.

### Related topics

- [Native Resource Manager Overview](#)
- [Appendix O: Resource Manager Integration](#)

# Managing Networks

## Network Management Overview

Network resources can be tightly integrated with the rest of a compute cluster using the Moab multi-resource manager management interface. This interface has the following capabilities:

- Dynamic per job and per partition [VLAN](#) creation and management
- Monitoring and reporting of network health and failure events
- Monitoring and reporting of network load
- Creation of subnets with guaranteed performance criteria
- Automated workload-aware configuration and router maintenance
- Intelligent network-aware scheduling algorithms

## Dynamic VLAN Creation

Most sites using dynamic VLAN's operate under the following assumptions:

- Each compute node has access to two or more networks, one of which is the compute network, and another which is the administrator network.
- Each compute node may only access other compute nodes via the compute network.
- Each compute node may only communicate with the head node via the administrator network.
- Logins on the head node may not be requested from a compute node.

In this environment, organizations may choose to have VLANs automatically configured that encapsulate individual jobs. These VLAN's essentially disconnect the job from either incoming or outgoing communication with other compute nodes.

## Configuring VLANs

Automated VLAN management can be enabled by setting up a network resource manager that supports dynamic VLAN configuration and a QoS to request this feature. The example configuration highlights this setup:

```
...
RMCFG[cisco] TYPE=NATIVE RESOURCETYPE=NETWORK FLAGS=VLAN
RMCFG[cisco] CLUSTERQUERYURL=exec://$TOOLSDIR/node.query.cisco.pl
RMCFG[cisco] SYSTEMMODIFYURL=exec://$TOOLSDIR/system.modify.cisco.pl
QOSCFG[netsecure] SECURITY=VLAN
```

## Requesting a VLAN

VLANs can be requested on a per job basis directly using the associated resource manager extension or indirectly by requesting a QoS with a VLAN security requirement.

```
> qsub -l nodes=256,walltime=24:00:00,qos=netsecure biojob.cmd
143325.umc.com submitted
```

## Network Load and Health Monitoring

Network-level load and health monitoring is enabled by supporting the cluster query action in the network resource manager and specifying the appropriate **CLUSTERQUERYURL** attribute in the associated resource manager interface. Node (virtual node) query commands ([mnodectl](#), [checknode](#)) can be used to view this load and health information that will also be correlated with associated workload and written to persistent accounting records. Network load and health based event information can also be fed into [generic events](#) and used to drive appropriate event based [triggers](#).

At present, load and health attributes such as fan speed, temperature, port failures, and various core switch failures can be monitored and reported. Additional failure events are monitored and reported as support is added within the network management system.

## Providing Per-QoS and Per-Job Bandwidth Guarantees

Intra-job bandwidth guarantees can be requested on a per job basis using the [BANDWIDTH](#) resource manager extensions. If specified, Moab does not allow a job to start unless these criteria can be satisfied via proper resource allocation or dynamic network partitions. As needed, Moab makes future resource reservations to be able to guarantee required allocations.

**Example 3-147:**

```
> qsub -l nodes=24,walltime=8:00:00,bandwidth=1000 hex3chem.cmd
job 44362.qjc submitted
```

**i** If dynamic network partitions are enabled, a **NODEMODIFYURL** attribute must be properly configured to drive the network resource manager. See [Native Resource Manager Overview](#) for details.

## Enabling Workload-Aware Network Maintenance

Network-aware maintenance is enabled by supporting the modify action in the network resource manager and specifying the appropriate **NODEMODIFYURL** attribute in the associated resource manager interface. Administrator resource management commands, ([mnodectl](#) and [mrmctl](#)), will then be routed directly through the resource manager to the network management system. In addition, reservation and real-time generic event and generic metric [triggers](#) can be configured to intelligently drive these facilities for maintenance and auto-recovery purposes.

Maintenance actions can include powering on and off the switch as well as rebooting/recycling all or part of the network. Additional operations are enabled as supported by the underlying networks.

## Creating a Resource Management Interface for a New Network

Many popular networks are supported using interfaces provided in the Moab `tools` directory. If a required network interface is not available, a new one can be created using the following guidelines:

### General Requirements

In all cases, a network resource manager should respond to a cluster query request by reporting a single node with a node name that will not conflict with any existing compute nodes. This node should report as a minimum the **state** attribute.

### Monitoring Load

Network load is reported to Moab using the generic resource bandwidth. For greatest value, both configured and used bandwidth (in megabytes per second) should be reported as in the following example:

```
force10 state=idle ares=bandwidth:5466 cres=bandwidth:10000
```

### Monitoring Failures

Network warning and failure events can be reported to Moab using the **gevent** metric. If automated responses are enabled, embedded epochtime information should be included.

```
force10 state=idle gevent[checksum]='ECC failure detected on port 13'
```

## Controlling Router State

Router power state can be controlled as a system modify interface is created that supports the commands on, off, and reset.

## Creating VLANs

VLAN creation, management, and reporting is more advanced requiring persistent VLAN ID tracking, global pool creation, and other features. Use of existing routing interface tools as templates is highly advised. VLAN management requires use of both the cluster query interface and the system modify interface.

## Per-Job Network Monitoring

It is possible to gather network usage on a per job basis using the [Native](#) Interface. When the native interface has been configured to report [netin](#) and [netout](#) Moab automatically gathers this data through the life of a job and reports total usage statistics upon job completion.

```
...
node99 netin=78658 netout=1256
...
```

This information is visible to users and administrators via command-line utilities, the web portal, and the desktop graphical interfaces.

### Related topics

- [Native Resource Manager Overview](#)
- [Network Utilization Statistics](#)

## Intelligent Platform Management Interface

- [IPMI Overview](#)
- [Node IPMI Configuration](#)
- [Installing IPMItool](#)
- [Setting-up the BMC-Node Map File](#)
- [Configuring Moab's IPMI Tools](#)
- [Configuring Moab](#)
- [Ensuring Proper Setup](#)

## IPMI Overview

The Intelligent Platform Management Interface (IPMI) specification defines a set of common interfaces system administrators can use to monitor system health and manage the system. The IPMI interface can monitor temperature and other sensor information, query platform status and power-on/power-off compute nodes. As IPMI operates independently of the node's OS interaction with the node can happen

even when powered down. Moab can use IPMI to monitor temperature information, check power status, power-up, power-down, and reboot compute nodes.

## Node IPMI Configuration

IPMI must be enabled on each node in the compute cluster. This is usually done either through the node's BIOS or by using a boot CD containing IPMI utilities provided by the manufacturer. With regard to configuring IPMI on the nodes, be sure to enable IPMI-over-LAN and set a common login and password on all the nodes. Additionally, you must set a unique IP address for each node's BMC. Take note of these addresses as you will need them when reviewing the [Creating the IPMI BMC-Node Map File](#) section.

## Installing IPMITool

[IPMITool](#) is an open-source tool used to retrieve sensor information from the IPMI Baseboard Management Controller (BMC) or to send remote chassis power control commands. The IPMITool developer provides Fedora Core binary packages as well as a source tarball on the [IPMITool download page](#). Download and install IPMITool on the Moab head node and make sure the `ipmitool` binary is in the current shell PATH.

Proper IPMI setup and IPMITool configuration can be confirmed by issuing the following command on the Moab head node.

```
> ipmitool -I lan -U username -P password -H BMC IP chassis status
```

The output of this command should be similar to the following.

```
System Power          : off
Power Overload        : false
Power Interlock       : inactive
Main Power Fault      : false
Power Control Fault   : false
Power Restore Policy  : previous
Last Power Event      :
Chassis Intrusion     : inactive
Front-Panel Lockout   : inactive
Drive Fault           : false
Cooling/Fan Fault     : false
```

## Creating the IPMI BMC-Node Map File [OPTIONAL]

Since the BMC can be controlled via LAN, it is possible for the BMC to have its own unique IP address. Since this IP address is separate from the IP address of the node, a simple mapping file is required for Moab to know each node's BMC address. The file is a flat text file and should be stored in the Moab home directory. If a mapping file is needed, specify the name in the `config.ipmi.pl` configuration file in the `etc/` directory. The following is an example of the mapping file:

```
#<NodeID> <BMC IP>
node01 10.10.10.101
node02 10.10.10.102
node03 10.10.10.103
node04 10.10.10.104
node05 10.10.10.105
# NodeID = the name of the nodes returned with "mdiag -n"
# BMC IP = the IP address of the IPMI BMC network interface
```

Note that only the nodes specified in this file are queried for IPMI information. Also note that the mapping file is disabled by default and the nodes that are returned from Moab with `mdiag -n` are the ones that are queried for IPMI sensor data.

## Configuring the Moab IPMI Tools

The `tools/` subdirectory in the install directory already contains the Perl scripts needed to interface with IPMI. The following is a list of the Perl scripts that should be in the `tools/` directory; confirm these are present and executable.

```
ipmi.mon.pl      # The daemon front-end called by Moab
ipmi.power.pl   # The power control script called by Moab
__mon.ipmi.pl   # The IPMI monitor daemon that updates and caches IPMI data from nodes
```

Next, a few configuration settings need to be adjusted in the `config.ipmi.pl` file found in the `etc` subdirectory. The IPMI-over-LAN username and password need to be set to the values that were set in the [Node IPMI Configuration](#) section. Also, the IPMI query daemon's polling interval can be modified by adjusting `$pollInterval`. This specifies how often the IPMI-enabled nodes are queried to retrieve sensor data.

## Configuring Moab

To allow Moab to use the IPMI tools, a native resource manager is configured. To do this, the following lines must be added to `moab.cfg`:

```
...
# IPMI - Node monitor script
RMCFG[ipminative] TYPE=NATIVE CLUSTERQUERYURL=exec://$TOOLSDIR/ipmi.mon.pl
...
```

Next, the following lines can be added to allow Moab to issue IPMI power commands.

```
...
# IPMI - Power on/off/reboot script
RMCFG[ipminative] NODEPOWERURL=exec://$TOOLSDIR/ipmi.power.pl
...
```

Moab can be configured to perform actions based on sensor data. For example, Moab can shut down a compute node if its CPU temperature exceeds 100 degrees Celsius, or it can power down idle compute nodes if workload is low. Generic event thresholds are used to tell Moab to perform certain duties given certain conditions. The following example is of a way for Moab to recognize it should power off a compute node if its CPU0 temperature exceeds 100 degrees Celsius.

```
...
# IPMI - Power off compute node if its CPU0 temperature exceeds 100 degrees Celsius.
GEVENTCFG[CPU0_TEMP>100] action=off
...
```

## Ensuring Proper Setup

Once the preceding steps have been taken, Moab should be started as normal. The IPMI monitoring daemon should start automatically, which can be confirmed with the following:

```
moab@headnode:~/ $ ps aux | grep _mon
moab 11444 0.0 0.3 6204 3172 pts/3 S 10:54 0:00 /usr/bin/perl -w
/opt/moab/tools/_mon.ipmi.pl --start
```

After a few minutes, IPMI data should be retrieved and cached. This can be confirmed with the following command:

```
moab@headnode:~/ $ cat spool/ipmicache.gm
node01 GMETRIC[CPU0_TEMP]=49
node01 GMETRIC[CPU1_TEMP]=32
node01 GMETRIC[SYS_TEMP]=31
node01 POWER=ON
```

Finally, issue the following to ensure Moab is grabbing the IPMI data. Temperature data should be present in the Generic Metrics row.

```
moab@headnode:~/ $ checknode node01
node node01
State: Idle (in current state for 00:03:12)
Configured Resources: PROCS: 1 MEM: 2000M SWAP: 3952M DISK: 1M
Utilized Resources: ---
Dedicated Resources: ---
Generic Metrics: CPU0_TEMP=42.00,CPU1_TEMP=30.00,SYS_TEMP=29.00
...
```

## Resource Manager Translation

- [Translation Overview](#)
- [Translation Enablement Steps](#)

### Translation Overview

Resource manager translation allows end-users to continue to use existing job command scripts and familiar job management and resource query commands. This is accomplished by emulating external commands, routing the underlying queries to Moab, and then formatting the responses in a familiar manner. Using translation, job submission clients, job query clients, job control clients, and resource query clients can be emulated making switching from one resource manager to another transparent and preserving investment in legacy scripts, tools, and experience.

## Translation Enablement Steps

To enable translation, you must:

- Edit the Moab tools configuration file.
- Copy, rename, and link the emulation scripts to a shorter, easier-to-use name.

### Configure Translation Tools

Located in the `$MOABHOMEDIR/etc` directory are tools-specific configuration files. For each resource manager that has installed translation tools, edit the Moab tools configuration file in the `etc` directory. For example, if enabling LSF translation, do the following:

```
> vi $MOABHOMEDIR/etc/config.moab.pl
# Set the PATH to include directories for moab client commands - mjobctl, etc.
$ENV{PATH} = "/opt/moab/bin:$ENV{PATH}";
```

### Add Translation Tools

In a directory accessible to users, create links to (or copy) the emulation scripts you want your users to use. For example, the emulation script `tools/bjobs.lsf.pl` could be copied to `bin/bjobs`, or, a symbolic link could be created in `bin/bjobs` that points to `tools/bjobs.lsf.pl`.

```
> ln -s tools/bjobs.lsf.pl bin/bjobs
> ln -s tools/bhosts.lsf.pl bin/bhosts
```

## Troubleshooting and System Maintenance

- [Internal Diagnostics/Diagnosing System Behavior and Problems on page 673](#)
- [Logging Overview on page 676](#)
- [Object Messages on page 682](#)
- [Notifying Administrators of Failures on page 684](#)
- [Issues with Client Commands on page 685](#)
- [Tracking System Failures on page 686](#)
- [Problems with Individual Jobs on page 688](#)
- [Diagnostic Scripts on page 689](#)

### Internal Diagnostics/Diagnosing System Behavior and Problems

Moab provides a number of commands for diagnosing system behavior. These diagnostic commands present detailed state information about various aspects of the scheduling problem, summarize

performance, and evaluate current operation reporting on any unexpected or potentially erroneous conditions found. Where possible, Moab's diagnostic commands even correct detected problems if desired.

At a high level, the diagnostic commands are organized along functionality and object based delineations. Diagnostic commands exist to help prioritize workload, evaluate fairness, and determine effectiveness of scheduling optimizations. Commands are also available to evaluate reservations reporting state information, potential reservation conflicts, and possible corruption issues. Scheduling is a complicated task. Failures and unexpected conditions can occur as a result of resource failures, job failures, or conflicting policies.

Moab's diagnostics can intelligently organize information to help isolate these failures and allow them to be resolved quickly. Another powerful use of the diagnostic commands is to address the situation in which there are no hard failures. In these cases, the jobs, compute nodes, and scheduler are all functioning properly, but the cluster is not behaving exactly as desired. Moab diagnostics can help a site determine how the current configuration is performing and how it can be changed to obtain the desired behavior.

## The mdiag Command

The cornerstone of Moab's diagnostics is the [mddiag](#) command. This command provides detailed information about scheduler state and also performs a large number of internal sanity checks presenting problems it finds as warning messages.

Currently, the `mddiag` command provides in-depth analysis of the following objects and subsystems:

Object/Subsystem	mddiag Flag	Use
<b>Account</b>	-a	Shows detailed account configuration information.
<b>Blocked</b>	<a href="#">-b</a>	Indicates why blocked (ineligible) jobs are not allowed to run.
<b>Class</b>	-c	Shows detailed class configuration information.
<b>Config</b>	-C	Shows configuration lines from <code>moab.cfg</code> and whether or not they are valid.
<b>FairShare</b>	<a href="#">-f</a>	Shows detailed fairshare configuration information as well as current fair-share usage.
<b>Group</b>	-g	Shows detailed group information.
<b>Job</b>	<a href="#">-j</a>	Shows detailed job information. Reports corrupt job attributes, unexpected states, and excessive job failures.
<b>Frame/Rack</b>	-m	Shows detailed frame/rack information.

Object/Subsystem	mdiag Flag	Use
<b>Node</b>	-n	Shows detailed node information. Reports unexpected node states and resource allocation conditions.
<b>Priority</b>	<u>-p</u>	Shows detailed job priority information including priority factor contributions to all idle jobs.
<b>QoS</b>	-q	Shows detailed QoS information.
<b>Reservation</b>	<u>-r</u>	Shows detailed reservation information. Reports reservation corruption and unexpected reservation conditions.
<b>Resource Manager</b>	<u>-R</u>	Shows detailed resource manager information. Reports configured and detected state, configuration, performance, and failures of all configured resource manager interfaces.
<b>Standing Reservations</b>	<u>-s</u>	Shows detailed standing reservation information. Reports reservation corruption and unexpected reservation conditions.
<b>Scheduler</b>	<u>-S</u>	Shows detailed scheduler state information. Indicates if scheduler is stopped, reports status of grid interface, and identifies and reports high-level scheduler failures.
<b>Partition</b>	-t	Shows detailed partition information.
<b>User</b>	<u>-u</u>	Shows detailed user information.

## Other Diagnostic Commands

Beyond `mdiag`, the [checkjob](#) and [checknode](#) commands also provide detailed information and sanity checking on individual jobs and nodes respectively. These commands can indicate why a job cannot start, which nodes can be available, and information regarding the recent events impacting current job or nodes state.

## Using Moab Logs for Troubleshooting

Moab logging is extremely useful in determining the cause of a problem. Where other systems may be cursed for not providing adequate logging to diagnose a problem, Moab may be cursed for the opposite reason. If the logging level is configured too high, huge volumes of log output may be recorded, potentially obscuring the problems in a flood of data. Intelligent searching combined with the use of the [LOGLEVEL](#) and [LOGFACILITY](#) parameters can mine out the needed information. Key information associated with various problems is generally marked with the keywords WARNING, ALERT, or ERROR. See the [Logging Overview](#) for further information.

## Automating Recovery Actions after a Failure

The [RECOVERYACTION](#) parameter of [SCHEDCFG](#) can be used to control scheduler action in the case of a catastrophic internal failure. Valid actions include die, ignore, restart, and trap.

Recovery Mode	Description
<b>die</b>	Moab will exit and, if core files are externally enabled, create a core file for analysis (This is the default behavior.).
<b>ignore</b>	Moab will ignore the signal and continue processing. This may cause Moab to continue running with corrupt data which may be dangerous. Use this setting with caution.
<b>restart</b>	When a SIGSEGV is received, Moab will relaunch using the current checkpoint file, the original launch environment, and the original command line flags. The receipt of the signal will be logged but Moab will continue scheduling. Because the scheduler is restarted with a new memory image, no corrupt scheduler data should exist. One caution with this mode is that it may mask underlying system failures by allowing Moab to overcome them. If used, the event log should be checked occasionally to determine if failures are being detected.
<b>trap</b>	When a SIGSEGV is received, Moab stays alive but enters diagnostic mode. In this mode, Moab stops scheduling but responds to client requests allowing analysis of the failure to occur using internal diagnostics available via the <a href="#">mdiag</a> command.

### Related topics

- [Troubleshooting Individual Jobs](#)

## Logging Overview

The Moab Workload Manager provides the ability to produce detailed logging of all of its activities. This is accomplished using verbose server logging, event logging, and system logging facilities.

- [Log Facility Configuration](#) on page 677
- [Standard Log Format](#) on page 678
- [Searching Moab Logs](#) on page 679
- [Event Logs](#) on page 679
  - [Event Log Format](#) on page 679
  - [Exporting Events in Real-Time](#) on page 681
- [Logging Overview](#) on page 676
- [Enabling Syslog](#) on page 681
- [Managing Verbosity](#) on page 682

## Log Facility Configuration

The [LOGFILE](#) and/or [LOGDIR](#) parameters within the `moab.cfg` file specify the destination of this logging information. Logging information will be written in the file `<MOABHOMEDIR>/<LOGDIR><LOGFILE>` unless `<LOGDIR>` or `<LOGFILE>` is specified using an absolute path. If the log file is not specified or points to an invalid file, all logging information is directed to `STDERR`. However, because of the sheer volume of information that can be logged, it is not recommended that this be done while in production. By default, [LOGDIR](#) and [LOGFILE](#) are set to `log` and `moab.log` respectively, resulting in scheduler logs being written to `<MOABHOMEDIR>/log/moab.log`.

The parameter [LOGFILEMAXSIZE](#) determines how large the log file is allowed to become before it is rolled and is set to 10 MB by default. When the log file reaches this specified size, the log file is rolled. The parameter [LOGFILEROLLDEPTH](#) controls the number of old logs maintained and defaults to 3. Rolled log files have a numeric suffix appended indicating their order.

The parameter [LOGLEVEL](#) controls the verbosity of the information. [LOGLEVEL](#) values between 1 and 6 are used to control the amount of information logged with 1 being the least verbose (recording only the worst events that occur) while 6 is the most verbose. The amount of information provided at each log level is approximately an order of magnitude greater than what is provided at the log level immediately below it. The first three log levels (1-3) measure the severity of an event and the rest of the levels (4-6) measure verbosity and how much detail is logged.

If a problem is detected, you may want to increase the [LOGLEVEL](#) value to get more details. However, doing so will cause the logs to roll faster and will also cause a lot of possibly unrelated information to clutter up the logs. Also be aware of the fact that high [LOGLEVEL](#) values results in large volumes of possibly unnecessary file I/O to occur on the scheduling machine. Consequently, it is not recommended that high [LOGLEVEL](#) values be used unless tracking a problem or similar circumstances warrant the I/O cost.



If high log levels are desired for an extended period of time and your Moab home directory is located on a network file system, performance may be improved by moving your log directory to a local file system using the [LOGDIR](#) parameter.

Visibility	LOGLEVEL value	Description
FATAL	n/a	FATAL events are errors that render part of the system unusable. An example would be failing to create a connection to a database. FATAL event logging cannot be suppressed.
ERROR	1	This is the minimum level of logging that Moab accepts. ERROR events are problems that occur in circumstances where a user's goal has failed. For example, when a user submits a job but the job fails to start, the cause of the failure will be an error. Not all failures are ERROR events, such as failing to open a file because it does not exist. Like FATAL events, ERROR events cannot be suppressed.

Visibility	LOGLEVEL value	Description
WARNING	2	WARNING events are problems that have user consequences that Moab cannot easily evaluate. Their impact has to be judged by users. An example would be if a user job asked Moab to copy a folder and Moab was unable to copy one file in the folder because the file was a temp file and was opened exclusively by another process. The user might consider that failure irrelevant. WARNING event logging can be suppressed at user discretion.
INFO	3	INFO events are occurrences that might be interesting but do not represent problems. An example would be the transition to a "terminated phase" when a service is successfully terminated. This event is potentially interesting to both human and automated observers but is not a problem in any sense.
TRACE1	4	These log levels are generally not used in production environments. They are used mainly by Adaptive Computing developers to analyze various issues. Setting LOGLEVEL to one of these levels could seriously impact performance due to Moab attempting to write to the log potentially hundreds of times per second.
TRACE2	5	
TRACE3	6	

A final log related parameter is **LOGFACILITY**. This parameter can be used to focus logging on a subset of scheduler activities. This parameter is specified as a list of one or more scheduling facilities as listed in the parameters documentation.

Example 3-148:

```
# moab.cfg
# allow up to 30 100MB logfiles
LOGLEVEL          3
LOGDIR            /var/tmp/moab
LOGFILEMAXSIZE   100000000
LOGFILEROLLDEPTH 30
```

## Standard Log Format

Each log event line follows a standard, tab-delimited log format:

**timestamp** <tab> **thread ID** <tab> **visibility** <tab> **origin** <tab> **event code** <tab> **scope IDs** <tab> **message**

- **timestamp:** Timestamps are given in local time, in [ISO 8601 format](#), with a 4-digit timezone offset suffix. For example, 2012-01-27T15:18:30.000-0700.
- **thread ID:** The ID of the thread that is producing the log output.
- **visibility:** Visibility is either a severity (FATAL, ERROR, WARNING, INFO) or a trace level (TRACE1, TRACE2, TRACE3).
- **origin:** Origin is where the log event came from.

- **event code:** The event code provides a way to determine what kind of event happened. For a full list of event codes, see [Event Dictionary on page 1098](#).
- **scope IDs:** The scope ID associates the event with a specific job or service.
- **message:** Messages can give details about the event and possibly some action information to resolve issues.

## Searching Moab Logs

While major failures are reported via the `mdiag -S` command, these failures can also be uncovered by searching the logs using the `grep` command as in the following:

```
> grep -E "WARNING|ALERT|ERROR" moab.log
```

On a production system working normally, this list usually includes some ALERT and WARNING messages. The messages are usually self-explanatory, but if not, viewing the log can give context to the message.

If a problem is occurring early when starting the Moab scheduler (before the configuration file is read) Moab can be started up using the `-L <LOGLEVEL>` flag. If this is the first flag on the command line, then the **LOGLEVEL** is set to the specified level immediately before any setup processing is done and additional logging is recorded.

If problems are detected in the use of one of the client commands, the client command can be re-issued with the `--loglevel=<LOGLEVEL>` command line argument specified. This argument causes log information to be written to STDERR as the client command is running. As with the server, `<LOGLEVEL>` values from 0 to 9 are supported.

The **LOGLEVEL** can be changed dynamically by use of the `mschedctl -m` command, or by modifying the `moab.cfg` file and restarting the scheduler. Also, if the scheduler appears to be hung or is not properly responding, the log level can be incremented by one by sending a **SIGUSR1** signal to the scheduler process. Repeated **SIGUSR1** signals continue to increase the log level. The **SIGUSR2** signal can be used to decrease the log level by one.

If an unexpected problem does occur, save the log file as it is often very helpful in isolating and correcting the problem.

## Event Logs

Major events are reported to both the Moab log file as well as the Moab event log. By default, the event log is maintained in the statistics directory and rolls on a daily basis, using the naming convention `events.WWW_MMM_DD_YYYY` as in `events.Tue_Mar_18_2008`.

### Event Log Format

The event log contains information about major job, reservation, node, and scheduler events and failures and reports this information in the following format:

```
<EVENTTIME> <EPOCHTIME>:<EID> <OBJECT> <OBJECTID> <EVENT> <DETAILS>
```

Example 3-149:

```

VERSION 500
07:03:21 110244322:0 sched clusterA start
07:03:26 110244327:1 rsv system.1 start 1124142432 1324142432 2 2 0.0 2342155.3
node1|node2 NA RSV=%=system.1=
07:03:54 110244355:2 job 1413 end 8 16 llw mcc 432000 Completed [batch:1]
11 08708752 1108703981 ...
07:04:59 110244410:3 rm base failure cannot connect to RM
07:05:20 110244431:4 sched clusterA stop admin
...
    
```

The parameter [RECORDEVENTLIST](#) can be used to control which events are reported to the event log. See the sections on [job](#) and [reservation](#) trace format for more information regarding the values reported in the details section for those records.

### Record Type Specific Details Format

The format for each record type is unique and is described in the following table:

Record Type	Event Types	Description
<b>gevent</b>	See <a href="#">Enabling Generic Events</a> for gevent information.	 Generic events are included within node records. See <a href="#">node</a> detail format that follows.
<b>job</b>	<i>JOBCANCEL, JOBCHECKPOINT, JOBEND, JOBHOLD, JOBMIGRATE, JOBMODIFY, JOBPREEMPT, JOBREJECT, JOBRESUME, JOBSTART, JOBSUBMIT</i>	See <a href="#">Workload Accounting Records</a> .
<b>node</b>	<i>NODEDOWN, NODEFAILURE, NODEUP</i>	The following fields are displayed in the event file in a space-delimited line as long as Moab has information pertaining to it: state, partition, disk, memory, maxprocs, swap, os, rm, nodeaccesspolicy, class, and message, where state is the node's current state and message is a human readable message indicating reason for node state change.
<b>rm</b>	<i>RMDOWN, RMPOLLEND, RMPOLLSTART, RMUP</i>	Human readable message indicating reason for resource manager state change.  For <i>SCHEDCOMMAND</i> , only create/modify commands are recorded. No record is created for general list/query commands. <i>ALLSCHEDCOMMAND</i> does the same thing as <i>SCHEDCOMMAND</i> , but it also logs info query commands.

Record Type	Event Types	Description
<b>trigger</b>	<i>TRIGEND, TRIGFAILURE, TRIGSTART</i>	<p>&lt;ATTR&gt;=&lt;VALUE&gt; [ &lt;ATTR&gt;=&lt;VALUE&gt; ] . . .</p> <p>where &lt;ATTR&gt; is one of the following: actiondata, actiontype, description, ebuf, eventtime, eventtype, flags, name, objectid, object-type, obuf, offset, period, requires, sets, threshold, timeout, and so forth.</p> <p>See <a href="#">About object triggers on page 745</a> for more information.</p>
<b>vm</b>	<i>VMCREATE, VMDESTROY, VMMIGRATE, VMPOWEROFF, VMPOWERON</i>	The following fields are displayed in the event file in a space-delimited line as long as Moab has information pertaining to it: name, sovereign, powerstate, parentnode, swap, memory, disk, maxprocs, opsys, class, and variables, where class and variables may have 0 or multiple entries.

## Exporting Events in Real-Time

Moab event information can be exported to external systems in real-time using the [ACCOUNTINGINTERFACEURL](#) parameter. When set, Moab activates this URL each time one of the default events or one of the events specified by the [RECORDEVENTLIST](#) occurs.

While various protocols can be used, the most common protocol is `exec`, which indicates that Moab should launch the specified tool or script and pass in event information as command line arguments. This tool can then select those events and fields of interest and re-direct them as appropriate providing significant flexibility and control to the organization.

### Exec Protocol Format

When a URL with an `exec` protocol is specified, the target is launched with the event fields passed in as STDIN. These fields appear exactly as they do in the [event logs](#) with the same values and order.

 The `tools/sql` directory included with the Moab distribution contains `event.create.sql.pl`, a sample accounting interface processing script that may be used as a template.

## Enabling Syslog

In addition to the log file, the Moab scheduler can report events it determines to be critical to the UNIX syslog facility via the daemon facility using priorities ranging from `INFO` to `ERROR`. (See [USESYSLOG](#)). The verbosity of this logging is not affected by the [LOGLEVEL](#) parameter. In addition to errors and critical events, user commands that affect the state of the jobs, nodes, or the scheduler may also be logged to syslog. Moab syslog messages are reported using the `INFO`, `NOTICE`, and `ERR` syslog priorities.

By default, messages are logged to syslog's user facility. However, using the [USESYSLOG](#) parameter, Moab can be configured to use any of the following:

- *user*
- *daemon*

- [local0](#)
- [local1](#)
- [local2](#)
- [local3](#)
- [local4](#)
- [local5](#)
- [local6](#)
- [local7](#)

## Managing Verbosity

In very large systems, a highly verbose log may roll too quickly to be of use in tracking specific targeted behaviors. In these cases, one or more of the following approaches may be of use:

- Use the [LOGFACILITY](#) parameter to log only functions and services of interest.
- Use [syslog](#) to maintain a permanent record of critical events and failures.
- Specify higher object loglevels on jobs, nodes, and reservations of interest (such as `NODECFG [orion13] LOGLEVEL=6`).
- Increase the range of events reported to the event log using the [RECORDEVENTLIST](#) parameter.
- Review object messages for required details.
- Run Moab in [monitor](#) mode using [IGNOREUSERS](#), [IGNOREJOBS](#), [IGNORECLASSES](#), or [IGNORENODES](#).

### Related topics

- [RECORDEVENTLIST](#) parameter
- [USESYSLOG](#) parameter
- [Notifying Admins](#)
- [Simulation Workload Trace Overview](#)
- [mschedctl -L](#) command

## Object Messages

### Object Message Overview

Messages can be associated with the scheduler, jobs, and nodes. Their primary use is a line of communication between resource managers, the scheduler, and end-users. When a node goes offline, or when a job fails to run, both the resource manager and the scheduler will post messages to the object's message buffer, giving the administrators and end-users a reason for the failure. They can also be used as a way for different administrators and users to send messages associated with the various objects.

For example, an administrator can set the message `Node going down for maintenance Apr/6/08 12pm,` on node `node01`, which would then be visible to other administrators.

## Viewing Messages

To view messages associated with a job (either from users, the resource manager, or Moab), run the [checkjob](#) command.

To view messages associated with a node (either from users, the resource manager, or Moab), run the [checknode](#) command.

To view system messages, use the [mschedctl -l](#) message command.

To view the messages associated with a credential, run the [mcredctl -c](#) command.

## Creating Messages

To create a message use the [mschedctl -c message](#) *<STRING>* [-o *<OBJECTTYPE>*:*<OBJECTID>*] [-w *<ATTRIBUTE>*=*<VALUE>* [-w ... ] ] command.

The *<OBJECTTYPE>* can be one of the following:

- node
- job
- rsv
- user
- acct
- qos
- class
- group

The *<ATTRIBUTE>* can be one of the following:

- owner
- priority
- expiretime
- type

Valid types include:

- annotation
- other
- hold
- pendactionerror

## Deleting Messages

Deleting, or removing, messages is straightforward. The commands used depend on the type of object to which the message is attached:

- Scheduler: Use the "[mschedctl](#) -d message:<INDEX>" command (where INDEX is the index of the message you want to delete).
- Node: Use the [mnodectl](#)<NODE> -d message:<INDEX> command.

## Notifying Administrators of Failures

### Enabling Administrator Email

In the case of certain events, Moab can automatically send email to administrators. To enable mail notification, the [MAILPROGRAM](#) parameter must be set to *DEFAULT* or point to the locally available mail client. With this set, policies such as [JOBREJECTPOLICY](#) will send email to administrators if set to a value of *MAIL*.

### Handling Events with the Notification Routine

Moab possesses a primitive event management system through the use of the notify program. The program is called each time an event of interest occurs. Currently, most events are associated with failures of some sort but use of this facility need not be limited in this way. The [NOTIFICATIONPROGRAM](#) parameter allows a site to specify the name of the program to run. This program is most often locally developed and designed to take action based on the event that has occurred. The location of the notification program may be specified as a relative or absolute path. If a relative path is specified, Moab looks for the notification relative to the \$(INSTDIR)/tools directory. In all cases, Moab verifies the existence of the notification program at start up and disables it if it cannot be found or is not executable.

The notification program's action may include steps such as reporting the event via email, adjusting scheduling parameters, rebooting a node, or even recycling the scheduler.

For most events, the notification program is called with command line arguments in a simple <EVENTTYPE>: <MESSAGE> format. The following event types are currently enabled:

Event Type	Format	Description
<b>JOBCORRUPTION</b>	<MESSAGE>	An active job is in an unexpected state or has one or more allocated nodes that are in unexpected states.
<b>JOBHOLD</b>	<MESSAGE>	A job hold has been placed on a job.
<b>JOBWCVIOLATION</b>	<MESSAGE>	A job has exceeded its wallclock limit.
<b>RESERVATIONCORRUPTION</b>	<MESSAGE>	Reservation corruption has been detected.

Event Type	Format	Description
<b>RESERVATIONCREATED</b>	<RSVNAME> <RSVTYPE> <NAME> <PRESENTTIME> <STARTTIME> <ENDTIME> <NODECOUNT>	A new reservation has been created.
<b>RESERVATIONDESTROYED</b>	<RSVNAME> <RSVTYPE> <PRESENTTIME> <STARTTIME> <ENDTIME> <NODECOUNT>	A reservation has been destroyed.
<b>RMFAILURE</b>	<MESSAGE>	The interface to the resource manager has failed.

Perhaps the most valuable use of the notify program stems from the fact that additional notifications can be easily inserted into Moab to handle site specific issues. To do this, locate the proper block routine, specify the correct conditional statement, and add a call to the routine `notify(<MESSAGE>);`.

#### Related topics

- [JOBREJECTPOLICY](#) parameter
- [MAILPROGRAM](#) parameter
- [Event Log Overview](#)

## Issues with Client Commands

- [Client Overview](#)
- [Diagnosing Client Problems](#)

### Client Overview

Moab client commands are implemented as links to the executable `mclient`. When a Moab client command runs, the client executable determines the name under which it runs and behaves accordingly. At the time Moab was configured, a home directory was specified. The Moab client attempts to open the configuration file, `moab.cfg`, in the `etc/` folder of this home directory on the node where the client command executes. This means that the home directory specified at configure time must be available on all hosts where the Moab client commands are executed. This also means that a `moab.cfg` file must be available in the `etc/` folder of this home directory. When the clients open this file, they will try to load the **SCHEDCFG** parameter to determine how to contact the Moab server.



The home directory value specified at configure time can be overridden by creating an `/etc/moab.cfg` file or by setting the `MOABHOMEDIR` environment variable.

Once the client has determined where the Moab server is located, it creates a message, adds an encrypted checksum, and sends the message to the server. The Moab client and Moab server must use a

shared secret key for this to work. When the Moab server receives the client request and verifies the message, it processes the command and returns a reply.

## Diagnosing Client Problems

The easiest way to determine where client failures are occurring is to use built-in Moab logging. On the client side, use the `--loglevel` flag. For example:

```
> showq --loglevel=9
```

*This will display verbose logging information regarding the loading of the configuration file, connecting to the Moab server, sending the request, and receiving a response.*

This information almost always reveals the source of the problem. If it does not, the next step is to look at the Moab server side logs; this is done using the following steps:

- Stop Moab scheduling so that the only activity is handling Moab client requests.

```
> mschedctl -s
```

- Set the logging level to *very* verbose.

```
> mschedctl -m loglevel 7
```

- Watch Moab activity.

```
> tail -f log/moab.log | more
```

Now, in a second window, issue any failing client command, such as [showq](#).

The `moab.log` file will record the client request and any reasons it was rejected.

## Tracking System Failures

### System Failures

The scheduler has a number of dependencies that may cause failures if not satisfied. These dependencies are in the areas of disk space, network access, memory, and processor utilization.

#### Disk Space

The scheduler uses a number of files. If the file system is full or otherwise inaccessible, the following behaviors might be noted:

Unavailable File	Behavior
<code>moab.pid</code>	Scheduler cannot perform single instance check.

Unavailable File	Behavior
<b>moab.ck*</b>	Scheduler cannot store persistent record of reservations, jobs, policies, summary statistics, and so forth.
<b>moab.cfg</b> <b>/moab.dat</b>	Scheduler cannot load local configuration.
<b>log/*</b>	Scheduler cannot log activities.
<b>stats/*</b>	Scheduler cannot write job records.

**i** When possible, configure Moab to use local disk space for configuration files, statistics files, and logs files. If any of these files are located in a networked file system (such as NFS, DFS, or AFS) and the network or file server experience heavy loads or failures, Moab server may appear sluggish or unresponsive and client command may fail. Use of local disk space eliminates susceptibility to this potential issue.

## Network

The scheduler uses a number of socket connections to perform basic functions. Network failures may affect the following facilities.

Network Connection	Behavior
<b>scheduler client</b>	Scheduler client commands fail.
<b>resource manager</b>	Scheduler is unable to load/update information regarding nodes and jobs.
<b>allocation manager</b>	Scheduler is unable to validate account access or reserve/debit account balances.

## Memory

Depending on cluster size and configuration, the scheduler may require up to 120 MB of memory on the server host. If inadequate memory is available, multiple aspects of scheduling may be negatively affected. The scheduler log files should indicate if memory failures are detected and mark any such messages with the ERROR or ALERT keywords.

## Processor Utilization

On a heavily loaded system, the scheduler may appear sluggish and unresponsive. However, no direct failures should result from this slowdown. Indirect failures may include timeouts of peer services (such as the resource manager or allocation manager) or timeouts of client commands. All timeouts should be recorded in the scheduler log files.

## Internal Errors

The Moab scheduling system contains features to assist in diagnosing internal failures. If the scheduler exits unexpectedly, the scheduler logs may provide information regarding the cause. If no reason can be determined, use of a debugger may be required.

### Logs

The first step in any exit failure is to check the last few lines of the scheduler log. In many cases, the scheduler may have exited due to misconfiguration or detected system failures. The last few lines of the log should indicate why the scheduler exited and what changes would be required to correct the situation. If the scheduler did not intentionally exit, increasing the `LOGLEVEL` parameter to `7`, or higher, may help isolate the problem.

## Reporting Failures

If an internal failure is detected on your system, the information of greatest value to developers in isolating the problem will be the output of the `gdb` where subcommand and a printout of all variables associated with the failure. In addition, a level 7 log covering the failure can also help in determining the environment that caused the failure. If you encounter such and require assistance, please submit a ticket at the following address:

<http://www.adaptivecomputing.com/services/techsupport.php>



If you do not already have a support username and password, please create a free account [to request a support ticket](#)

## Problems with Individual Jobs

To determine why a particular job will not start, there are several helpful commands:

### [checkjob -v](#)

`checkjob` evaluates the ability of a job to start immediately. Tests include resource access, node state, job constraints (such as `startdate`, `taskspernode`, and `QoS`). Additionally, command line flags may be specified to provide further information.

Flag	Description
<code>-l &lt;POLICYLEVEL&gt;</code>	Evaluates impact of throttling policies on job feasibility.
<code>-n &lt;NODENAME&gt;</code>	Evaluates resource access on specific node.
<code>-r &lt;RESERVATION_LIST&gt;</code>	Evaluates access to specified reservations.

### [checknode](#)

Displays detailed status of node.

[mddiag -b](#)

Displays various reasons job is considered blocked or non-queued.

[mddiag -j](#)

Displays high level summary of job attributes and performs sanity check on job attributes/state.

[showbf -v](#)

Determines general resource availability subject to specified constraints.

## Diagnostic Scripts

Moab Workload Manager provides diagnostic scripts that can help aid in monitoring the state of the scheduler, resource managers, and other important components of the cluster software stack. These scripts can also be used to help diagnose issues that may need to be resolved with the help of Adaptive Computing support staff. This section introduces available diagnostic scripts.

### support-diag.py

The `support-diag.py` script has a two-fold purpose. First, it can be used by a Moab trigger or cron job to create a regular snapshot of the state of Moab. The script captures the output of several Moab diagnostic commands (such as `showq`, `mddiag -n`, and `mddiag -S`), gathers configuration/log files, and records pertinent operating system information. This data is then compressed in a time-stamped tarball for easy long-term storage.

Second, the script provides Adaptive Computing support personnel with a complete package of information that can be used to help diagnose configuration issues or system bugs. After capturing the state of Moab, the resulting tarball can be sent to your Adaptive Computing support contact for further diagnosis.

The script asks you for the trouble ticket number, `-t <TICKET#>`, or `-n`. If you chose to enter `-t <TICKET#>` the script uploads your support diagnostic output to Adaptive Computing Customer Support. The upload and ticket number request can be prevented using the `-n` option.

### Synopsis

```
support-diag.py [<options>]
```

### Arguments

Argument	Description
<code>-h, --help</code>	Show this help message and exit.
<code>-q, --diag-torque-off, --without-torque</code>	Disable TORQUE diagnostics.

Argument	Description
<code>-p TMPDIR, --tmp-dir=TMPDIR</code>	Use a different tmp directory to store output.
<code>-n, --no-upload</code>	Do not upload to Adaptive Computing.
<code>-t TICKET#</code>	Support ticket number.
<code>-f, --full-mode</code>	Gather additional logs, stats and, <code>moab.db</code> files.
<code>-u TIMEOUT, --moab-timeout=TIMEOUT</code>	Define Moab command timeout (Default 300 seconds).
<code>-d, --debug-mode</code>	<code>support-diag</code> print debug variables.
<code>-o, --offline-mode</code>	Gather offline logging only.
<code>-r, --ftp</code>	Use ftp instead of scp.
<code>-V, --version</code>	Print version information.

## support.diag.pl

 This script is deprecated with the 8.0 release. Use the `support-diag.py` script instead.

The `support.diag.pl` script has a two-fold purpose. First, it can be used by a Moab trigger or cron job to create a regular snapshot of the state of Moab. The script captures the output of several Moab diagnostic commands (such as `showq`, `mdiag -n`, and `mdiag -S`), gathers configuration/log files, and records pertinent operating system information. This data is then compressed in a time-stamped tarball for easy long-term storage.

The second purpose of the `support.diag.pl` script is to provide Adaptive Computing support personnel with a complete package of information that can be used to help diagnose configuration issues or system bugs. After capturing the state of Moab, the resulting tarball could be sent to your Adaptive Computing support contact for further diagnosis.

The `support.diag.pl` will ask you for the trouble ticket number then guide you through the process of uploading the data to Adaptive Computing Customer Support. The uploading and ticket number request may be prevented using the `--no-upload` and `--support-ticket=<SUPPORT_TICKET_ID>` flags detailed in the Arguments table that follows.

### Synopsis

```
support.diag.pl [--include-log-lines=<NUM>] [--diag-torque]
```

## Arguments

Argument	Description
<code>--include-log-lines=&lt;NUM&gt;</code>	Instead of including the entire <code>moab.log</code> file, only the last <code>&lt;NUM&gt;</code> lines are captured in the diagnostics.
<code>--diag-torque</code>	Diagnostic commands pertinent to the TORQUE resource manager are included.
<code>--no-upload</code>	Prevents the system from asking the user if they want to upload the tarball to Adaptive Computing Customer Support.
<code>--support-ticket=&lt;SUPPORT_TICKET_ID&gt;</code>	Prevents the system from asking the user for a support ticket number.

## Improving User Effectiveness

- [User Feedback Loops on page 691](#)
- [User Level Statistics on page 692](#)
- [Enhancing Wallclock Limit Estimates on page 693](#)
- [Job Start Time Estimates on page 693](#)
- [Providing Resource Availability Information on page 694](#)
- [Collecting Performance Information on Individual Jobs on page 694](#)

## User Feedback Loops

Almost invariably, real world systems outperform simulated systems, even when all policies, reservations, workload, and resource distributions are fully captured and emulated. What is it about real world usage that is not emulated via a simulation? The answer is the user feedback loop, the impact of users making decisions to optimize their level of service based on real time information.

A user feedback loop is created any time information is provided to a user that modifies job submission or job management behavior. As in a market economy, the cumulative effect of many users taking steps to improve their individual scheduling performance results in better job packing, lower queue time, and better overall system utilization. Because this behavior is beneficial to the system at large, system administrators and management should encourage this behavior and provide the best possible information to them.

There are two primary types of information that help users make improved decisions: cluster wide resource availability information and per job resource utilization information.

## Improving Job Size/Duration Requests

Moab provides a number of informational commands that help users make improved job management decisions based on real-time cluster wide resource availability information. These commands include [showbf](#), [showstats -f](#), and [showq](#). Using these commands, a user can determine what resources are available and what job configurations statistically receive the best scheduling performance.

## Improving Resource Requirement Specification

A job's resource requirement specification tells the scheduler what type of compute nodes are required to run the job. These requirements may state that a certain amount of memory is required per node or that a node has a minimum processor speed. At many sites, users will determine the resource requirements needed to run an initial job. Then, for the next several years, they will use the same basic batch command file to run all of their remaining jobs even though the resource requirements of their subsequent jobs may be very different from their initial run. Users often do not update their batch command files even though these constraints may be unnecessarily limiting the resources available to their jobs for two reasons: (1) users do not know how much their performance will improve if better information were provided and (2) users do not know exactly what resources their jobs are using and are afraid to lower their job's resource requirements since doing so might cause their job to fail.

To help with determining accurate per job resource utilization information, Moab provides the [FEEDBACKPROGRAM](#) facility. This tool allows sites to send detailed resource utilization information back to users via email, to store it in a centralized database for report preparation, or use it in other ways to help users refine their batch jobs.

## User Level Statistics

Besides displaying job queues, end-users can display a number of their own statistics. The [showstats -u <USER\\_ID>](#) command displays current and historical statistics for a user as seen in what follows:

```

$ showstats -u john
statistics initialized Wed Dec 31 17:00:00

      |----- Active -----|----- Completed -----
-----|
user      Jobs Procs ProcHours Jobs   %   PHReq   %   PHDed   %   FSTgt   AvgXF
MaxXF   AvgQH  Effic  WCAcc
john      1     1     30.96   9   0.00  300.0   0.00  148.9   0.00  -----  0.62
0.00     4.33 100.00  48.87
    
```

Users can query available system resources with the [showbf](#) command. This can aid users in requesting node configurations that are idle. Also, users can use the [checkjob](#) command to determine what parameter(s) are restricting their job from running. Moab performs better with more accurate wallclock estimates.

 Moab must use an ODBC-compliant database to report statistics with Viewpoint reports.

## Enhancing Wallclock Limit Estimates

As explained in the previous section, `showstats -u <USER_ID>` reports statistics for a given user. The `showstats -u` command can be accessed by all users. They can use fields such as `PHReq`, `PHDed`, or `WCAcc` to gauge wallclock estimates. Accurate wallclock estimates allow a job to be scheduled as soon as possible in a slot that it will fit in. Low or high estimates can cause a job to be scheduled in a less favorable position.

## Job Start Time Estimates

Each user can use the `showstart` command to display estimated start and completion times. The following example illustrates a typical response from issuing this command:

```
> showstart orion.13762
job orion.13762 requires 2 procs for 0:33:20
Estimated Rsv based start in           1:04:55 on Fri Jul 15 12:53:40
Estimated Rsv based completion in      2:44:55 on Fri Jul 15 14:33:40
Estimated Priority based start in      5:14:55 on Fri Jul 15 17:03:40
Estimated Priority based completion in  6:54:55 on Fri Jul 15 18:43:40
Estimated Historical based start in    00:00:00 on Fri Jul 15 11:48:45
Estimated Historical based completion in 1:40:00 on Fri Jul 15 13:28:45
Best Partition: fast
```

## Estimation Types

### Reservation-Based Estimates

Reservation-based start time estimation incorporates information regarding current administrative, user, and job reservations to determine the earliest time the specified job can allocate the needed resources and start running. In essence, this estimate indicates the earliest time the job will start, assuming this job is the highest priority job in the queue.

**i** For reservation-based estimates, the information provided by this command is more highly accurate if the job is highest priority, if the job has a reservation, or if the majority of the jobs that are of higher priority have reservations. Consequently, site administrators wanting to make decisions based on this information may want to consider using the `RESERVATIONDEPTH` parameter to increase the number of priority-based reservations. This can be set so that most, or even all, idle jobs receive priority reservations and make the results of this command generally useful. The only caution of this approach is that increasing the `RESERVATIONDEPTH` parameter more tightly constrains the decisions of the scheduler and may result in slightly lower system utilization (typically less than 8% reduction).

### Backlog/Priority Estimates

Priority-based job start analysis determines when the queried job will fit in the queue and determines the estimated amount of time required to complete the jobs currently running or scheduled to run before this job can start.

In all cases, if the job is running, this command returns the time the job starts. If the job already has a reservation, this command returns the start time of the reservation.

## Historical Estimates

Historical analysis uses historical queue times for jobs that match a similar processor count and job duration profile. This information is updated on a sliding window that is configurable within `moab.cfg`.

### Related topics

- [ENABLESTARTESTIMATESTATS](#) parameter
- [showstart](#) command

## Providing Resource Availability Information

Moab provides commands to allow the user to query available resources. The [showbf command](#) displays what resources are available for immediate use. Using different command line parameters, such as `-m`, `-n`, and `-q` allows the user to query resources based on memory, nodecount, or QoS respectively.

## Collecting Performance Information on Individual Jobs

Individual job information can be collected from the statistics file in [STATDIR](#), which contains start time, end time, end state, QoS requested, QoS delivered, and so forth for different jobs. Also, Moab optionally provides similar information to a site's feedback program. See section [21.1 User Feedback Overview](#) for more information about the feedback program.

## Cluster Analysis, Testing, and Simulation

- [Testing New Releases and Policies on page 695](#)
- [Testing New Middleware on page 699](#)

Moab has a number of unique features that allow site administrators to visualize current cluster behavior and performance, safely evaluate changes on production systems, and analyze probable future behaviors within a variety of environments.

These capabilities are enabled through a number of Moab facilities that may not appear to be closely related at first. However, taken together, these facilities allow organizations the ability to analyze their cluster without the losses associated with policy conflicts, unnecessary downtime, and faulty systems middleware.

Simulations allow organizations to evaluate many scenarios that could not be properly evaluated in real-world situations. In particular, these evaluations may be impossible due to time constraints, budgetary or personnel limitations, hardware availability, or even policy issues. In such cases, simulations provide information in countless scenarios and can help answer questions such as the following:

- What is the impact of additional hardware on cluster utilization?
- What delays to key projects can be expected with the addition of new users?
- How will new prioritization weights alter cycle distribution among existing workload?
- What total loss of compute resources will result from introducing a maintenance downtime?
- Are the benefits of cycle stealing from non-dedicated desktop systems worth the effort?
- How much will anticipated grid workload delay the average wait time of local jobs?

## Testing New Releases and Policies

- [Moab Evaluation Modes](#)
  - [MONITOR Mode](#)
  - [TEST Mode](#)
  - [INTERACTIVE Mode](#)
- [Testing New Releases](#)
- [Testing New Policies](#)
  - [Verifying Correct Specification of New Policies](#)
  - [Verifying Correct Behavior of New Policies](#)
  - [Determining Long Term Impact of New Policies](#)
- [Moab Side-by-Side](#)

## Moab Evaluation Modes

### MONITOR Mode

Moab supports a scheduling mode called *MONITOR*. In this mode, the scheduler initializes, contacts the resource manager and other peer services, and conducts scheduling cycles exactly as it would if running in *NORMAL* or production mode. Jobs are prioritized, reservations created, policies and limits enforced, and administrator and end-user commands enabled. The key difference is that although live resource management information is loaded, *MONITOR* mode disables Moab's ability to start, preempt, cancel, or otherwise modify jobs or resources. Moab continues to attempt to schedule exactly as it would in *NORMAL* mode, but its ability to actually impact the system is disabled. Using this mode, a site can quickly verify correct resource manager configuration and scheduler operation. This mode can also be used to validate new policies and constraints. In fact, Moab can be run in *MONITOR* mode on a production system while another scheduler or even another version of Moab is running on the same system. This unique ability can allow new versions and configurations to be fully tested without any exposure to potential failures and with no cluster downtime.

To run Moab in *MONITOR* mode, simply set the **MODE** attribute of the **SCHEDCFG** parameter to *MONITOR* and start Moab. Normal scheduler commands can be used to evaluate configuration and performance. [Diagnostic commands](#) can be used to look for any potential issues. Further, the Moab log

file can be used to determine which jobs Moab attempted to start, and which resources Moab attempted to allocate.

If another instance of Moab is running in production and a site administrator wants to evaluate an alternate configuration or new version, this is easily done but care should be taken to avoid conflicts with the primary scheduler. Potential conflicts include statistics files, logs, checkpoint files, and user interface ports. One of the easiest ways to avoid these conflicts is to create a new test directory with its own log and statistics subdirectories. The new `moab.cfg` file can be created from scratch or based on the existing `moab.cfg` file already in use. In either case, make certain that the **PORT** attribute of the **SCHEDCFG** parameter differs from that used by the production scheduler by at least two ports. If testing with the production binary executable, the `MOABHOMEDIR` environment variable should be set to point to the new test directory to prevent Moab from loading the production `moab.cfg` file.

## TEST Mode

*TEST* mode behaves much like *MONITOR* mode with the exception that Moab will log the scheduling actions it would have taken to the `stats/<DAY>.events` file. Using this file, sites can determine the actions Moab would have taken if running in *NORMAL* mode and verify all actions are in agreement with expected behavior.

## INTERACTIVE Mode

*INTERACTIVE* mode allows for evaluation of new versions and configurations in a manner different from *MONITOR* mode. Instead of disabling all resource and job control functions, Moab sends the desired change request to the screen and requests permission to complete it. For example, before starting a job, Moab may print something like the following to the screen:

```
Command: start job 1139.ncsa.edu on node list test013,test017,test018,test021
Accept: (y/n) [default: n]?
```

The administrator must specifically accept each command request after verifying it correctly meets desired site policies. Moab will then execute the specified command. This mode is highly useful in validating scheduler behavior and can be used until configuration is appropriately tuned and all parties are comfortable with the scheduler's performance. In most cases, sites will want to set the scheduling mode to *NORMAL* after verifying correct behavior.

## Testing New Releases

By default, Moab runs in a [mode](#) called *NORMAL*, which indicates that it is responsible for the cluster. It loads workload and resource information, and is responsible for managing that workload according to mission objectives and policies. It starts, cancels, preempts, and modifies jobs according to these policies.

If Moab is configured to use a mode called [TEST](#), it loads all information, performs all analysis, but, instead of actually starting or modifying a job, it merely logs the fact that it would have done so. A test instance of Moab can run at the same time as a production instance of Moab. A test instance of Moab can also run while a production scheduler of another type (such as PBS, LSF, or SLURM) is simultaneously running. This multi-scheduler ability allows stability and performance tests to be conducted that can help answer the following questions:

- What impact do Moab services have on network, processor, and memory load?
- What impact do Moab services have on the underlying resource manager?
- Is Moab able to correctly import resource, workload, policy, and credential information from the underlying resource manager?
- Are Moab's logged scheduling decisions in line with mission objectives?

In test mode, all of Moab's commands and services operate normally allowing the use of client commands to perform analysis. In most cases, the [mdiag](#) command is of greatest value, displaying loaded values as well as reporting detected failures, inconsistencies, and object corruption. The following table highlights the most common diagnostics performed.

Command	Object
<a href="#">mdiag -n</a>	Compute nodes, storage systems, network systems, and generic resources
<a href="#">mdiag -j</a>	Applications and static jobs
<a href="#">mdiag -u</a> <a href="#">mdiag -g</a> <a href="#">mdiag -a</a>	User, group, and account credentials
<a href="#">mdiag -c</a>	Queues and policies
<a href="#">mdiag -R</a>	Resource manager interface and performance
<a href="#">mdiag -S</a>	Scheduler/system level failures introduced by corrupt information

These commands will not only verify proper scheduling objects but will also analyze the behavior of each resource manager, recording failures, and delivered performance. If any misconfiguration, corruption, interface failure, or internal failure is detected, it can be addressed in the test mode instance of Moab with no urgency or risk to production cluster activities.

## Testing New Policies

### Verifying Correct Specification of New Policies

The first aspect of verifying a new policy is verifying correct syntax and semantics. If using [Moab Cluster Manager](#), this step is not necessary as this tool automatically verifies proper policy specification. If manually editing the `moab.cfg` file, the following command can be used for validation:

```
> mdiag -C
```

*This command will validate the configuration file and report any misconfiguration.*

## Verifying Correct Behavior of New Policies

If concern exists over the impact of a new policy, an administrator can babysit Moab by putting it into [INTERACTIVE](#) mode. In this mode, Moab will schedule according to all mission objectives and policies, but before taking any action, it will request that the administrator confirm the action. See the [interactive mode overview](#) for more information.

In this mode, only actions approved by the administrator will be carried out. Once proper behavior is verified, the Moab mode can be set to *NORMAL*.

## Determining Long Term Impact of New Policies

If a new policy has the potential to impact long-term performance or resource distribution, it may be desirable to run a Moab simulation to evaluate this change. Simulations allow locally recorded workload to be translated into simulation jobs and execute on a virtual cluster that emulates local resources. Simulations import all job and resource attributes that are loaded in a production environment as well as all policies specified in any configuration file. While running, all Moab commands and statistics are fully supported.

Using simulation, a control run can be made using the original policies and the behavior of this run compared to a second run that contains the specified change. Moab Cluster Manager's charting, graphing, and reporting features can be used to report on and visualize the differences in these two runs. Typically, a two-month real-time simulation can be completed in under an hour. For more information on simulations, see the Simulation Overview.

## Moab Side-by-Side

Moab provides an additional evaluation method that allows a production cluster or other resource to be logically partitioned along resource and workload boundaries and allows different instances of Moab to schedule different partitions. The parameters [IGNORENODES](#), [IGNORECLASSES](#), [IGNOREJOBS](#), and [IGNOREUSERS](#) are used to specify how the system is to be partitioned. In the following example, a small portion of an existing cluster is partitioned for temporary grid testing so that there is no impact on the production workload.

```
SCHEDCFG[prod] MODE=NORMAL SERVER=orion.cxz.com:42020
RMCFG[TORQUE]  TYPE=PBS
IGNORENODES    node61,node62,node63,node64
IGNOREUSERS    gridtest1,gridtest2
...
SCHEDCFG[prod] MODE=NORMAL SERVER=orion.cxz.com:42030
RMCFG[TORQUE]  TYPE=PBS
IGNORENODES    !node61,node62,node63,node64
IGNOREUSERS    !gridtest1,gridtest2
...
```

*Two completely independent Moab servers schedule the cluster. The first server handles all jobs and nodes except for the ones involved in the test. The second server handles only test nodes and test jobs. While both servers actively talk and interact with a single TORQUE resource manager, the **IGNORE\*** parameters cause them to not schedule, nor even see the other partition and its associated workload.*

- i** When enabling Moab side-by-side, each Moab server should have an independent home directory to prevent logging and statistics conflicts. Also, in this environment, each Moab server should communicate with its client commands using a different port as shown in the previous example.
- i** When specifying the **IGNORENODES** parameter, the exact node names, as returned by the resource manager, should be specified.

Related topics

- [Testing New Versions and Configurations](#)

## Testing New Middleware

Moab can be used to drive new middleware stress testing resource management systems, information services, allocation services, security services, data staging services, and other aspects. Moab is unique when compared to other stress testing tools as it can perform the tests in response to actual or recorded workload traces, performing a playback of events and driving the underlying system as if it were part of the production environment.

This feature can be used to identify scalability issues, pathological use cases, and accounting irregularities in anything from LDAP, to NIS, and NFS.

Using Moab's [time management](#) facilities, Moab can drive the underlying systems in accordance with the real recorded distribution of time, at a multiplier of real time, or as fast as possible.

The following table describes some aspects of cluster analysis that can be driven by Moab.

System	Details
<b>Allocation Manager</b>	Use <i>test</i> or <i>simulation</i> mode to drive scheduling queries, allocation debits, and reservations to accounting packages. Verify synchronization of cluster statistics and stress test interfaces and underlying databases.
<a href="#">On-Demand/Provisioning Services</a>	Use <i>simulation</i> or native resource manager mode to drive triggers and resource management interfaces to enable dynamic provisioning of hardware, operating systems, application software, and services. Test reliability and scalability of data servers, networks, and provisioning software as well as the interfaces and business logic coordinating these changes.
<a href="#">Resource Monitoring</a>	Use <i>test</i> or native resource manager mode to actively load information from compute, network, storage, and software license managers confirming validity of data, availability during failures, and scalability.

With each evaluation, the following tests can be enabled:

- functionality
- reliability
  - hard failure
    - hardware failure - compute, network, and data failures
    - software failure - loss of software services (NIS, LDAP, NFS, database)
    - soft failure
    - network delays, full file system, dropped network packets
  - corrupt data
- performance
- determine peak responsiveness in seconds/request
- determine peak throughput in requests/second
- determine responsiveness under heavy load conditions
- determine throughput under external load conditions
  - large user base (many users, groups, accounts)
  - large workload (many jobs)
  - large cluster (many nodes)
- manageability
  - full accounting for all actions/events
  - actions/failures can be easily and fully diagnosed

**i** If using a native resource manager and you do not want to actually submit real workload, you can set the environment variable `MFORCESUBMIT` to allow virtual workload to be managed without ever launching a real process.

## General Analysis

For all middleware interfaces, Moab provides built-in performance analysis and failure reporting. Diagnostics for these interfaces are available via the [mdia](#) command.

## Native Mode Analysis

Using [native mode](#) analysis, organizations can run Moab in *normal* mode with all facilities fully enabled, but with the resource manager fully emulated. With a native resource manager interface, any arbitrary cluster can be emulated with a simple script or flat text file. Artificial failures can be introduced, jobs can be virtually running, and artificial performance information generated and reported.

In the simplest case, emulation can be accomplished using the following configuration:

```
SCHEDCFG[natcluster] MODE=NORMAL SERVER=test1.bbli.com
ADMINCFG[1] USERS=dev
RMCFG[natcluster] TYPE=NATIVE CLUSTERQUERYURL=file://$HOME/cluster.dat
```

The preceding configuration will load cluster resource information from the file `cluster.dat`. An example resource information file follows:

```
node01 state=idle cproc=2
node02 state=idle cproc=2
node03 state=idle cproc=2
node04 state=idle cproc=2
node05 state=idle cproc=2
node06 state=idle cproc=2
node07 state=idle cproc=2
node08 state=idle cproc=2
```

In actual usage, any number of node attributes may be specified to customize these nodes, but in this example, only the node state and node configured processors attributes are specified.

The **RMCFG** flag *NORMSTART* indicates that Moab should not actually issue a job start command to an external entity to start the job, but rather start the job logically internally only.

If it is desirable to take an arbitrary action at the start of a job, end of a job, or anywhere in between, the **JOBCFG** parameter can be used to create one or more arbitrary [triggers](#) to initiate internal or external events. The triggers can do anything from executing a script, to updating a database, to using a Web service.

Using native resource manager mode, jobs may be introduced using the [msub](#) command according to any arbitrary schedule. Moab will load them, schedule them, and start them according to all site mission objectives and policies and drive all interfaced services as if running in a full production environment.

## Configuring Resources for Simulation

Resource traces fully describe all scheduling relevant aspects of a batch system's compute resources. In most cases, each resource trace describes a single compute node providing information about configured resources, node location, supported classes and queues, and so forth.

The resources Moab uses to simulate are created using Moab RM language. To create a Moab simulation this way, perform the following steps:

1. Load resources into Moab using a native RM. Create a file manually using the format for different resource attributes (see [W.1 Moab Resource Manager Language Data Format on page 1363](#)) or run `mnodectl -q wiki ALL > nodes.dat` to create a resources simulation file from the resources in your environment and write it to a file (in this example, a file called `nodes.dat`). You can modify the resource file to add resources, change attributes, etc.
2. Set up the resource manager interface by inserting the following into your `moab.cfg` file.

```
RMCFG[rmName] TYPE=NATIVE CLUSTERQUERYURL=FILE:///<locationOfFile>/<nameOfFile>
```

For the example in step 1, you would replace `<nameOfFile>` with `nodes.dat`.

3. Restart Moab.

## Sample Resource Trace

```
n8 STATE=Idle PARTITION=base AMEMORY=16000 APROC=4 CMEMORY=16000 CPROC=4 RM=base
NODEACCESSPOLICY=SHARED FEATURE=linux ARCH=x86_64
```

### Related topics

- [mnodectl -q wiki](#) - outputs Moab RM language format directly to a file

## Workload Event Format

Moab workload [accounting records](#) fully describe all scheduling relevant aspects of batch jobs including resources requested and used, time of all major scheduling events (such as submission time and start time), the job credentials used, and the job execution environment. Each job trace is composed of a single line consisting of whitespace delimited fields as shown in the following table.

**i** Moab can be configured to provide this information in flat text tabular form or in XML format conforming to the SSS 1.0 job description specification.

- [Workload Event Record Format](#)
- [Creating New Workload Accounting Records/Traces](#)
- [Reservation Records/Traces](#)
- [Recording Job events](#)

## Workload Event Record Format

All job events (*JOBSUBMIT*, *JOBSTART*, *JOBEND*, and so forth) provide job data in a standard format as described in the following table:

Field Name	Field Index	Data Format	Default Value	Details
<b>Event Time (Human Readable)</b>	1	HH:MM:SS	-	Specifies time event occurred.
<b>Event Time (Epoch)</b>	2	<epochtime>	-	Specifies time event occurred.
<b>Object Type</b>	3	job	-	Specifies record object type.
<b>Object ID</b>	4	<STRING>	-	Unique object identifier.

Field Name	Field Index	Data Format	Default Value	Details
<b>Object Event</b>	5	one of <i>job-cancel</i> , <i>jobcheckpoint</i> , <i>jobend</i> , <i>job-failure</i> , <i>jobhold</i> , <i>jobmigrate</i> , <i>job-preempt</i> , <i>jobreject</i> , <i>jobresume</i> , <i>jobstart</i> or <i>job-submit</i>	-	Specifies record <a href="#">event type</a> .
<b>Nodes Requested</b>	6	<INTEGER>	0	Number of nodes requested (0 = no node request count specified).
<b>Tasks Requested</b>	7	<INTEGER>	1	Number of tasks requested.
<b>User Name</b>	8	<STRING>	-	Name of user submitting job.
<b>Group Name</b>	9	<STRING>	-	Primary group of user submitting job.
<b>Wallclock Limit</b>	10	<INTEGER>	1	Maximum allowed job duration (in seconds).
<b>Job Event State</b>	11	<STRING>	-	Job <a href="#">state</a> at time of event.
<b>Required Class</b>	12	<STRING>	[DEFAULT:1]	<a href="#">Class/queue</a> required by job specified as square bracket list of <QUEUE> [:<QUEUEINSTANCE>] requirements. (For example: [batch:1]).
<b>Submission Time</b>	13	<INTEGER>	0	Epoch time when job was submitted.
<b>Dispatch Time</b>	14	<INTEGER>	0	Epoch time when scheduler requested job begin executing.
<b>Start Time</b>	15	<INTEGER>	0	Epoch time when job began executing. This is usually identical to Dispatch Time.

Field Name	Field Index	Data Format	Default Value	Details
<b>Completion Time</b>	16	<INTEGER>	0	Epoch time when job completed execution.
<b>Required Node Architecture</b>	17	<STRING>	-	Required node architecture if specified.
<b>Required Node Operating System</b>	18	<STRING>	-	Required node operating system if specified.
<b>Required Node Memory Comparison</b>	19	one of >, >=, =, <=, <	>=	Comparison for determining compliance with required node memory.
<b>Required Node Memory</b>	20	<INTEGER>	0	Amount of required configured RAM (in MB) on each node.
<b>Required Node Disk Comparison</b>	21	one of >, >=, =, <=, <	>=	Comparison for determining compliance with required node disk.
<b>Required Node Disk</b>	22	<INTEGER>	0	Amount of required configured local disk (in MB) on each node.
<b>Required Node Attributes/Features</b>	23	<STRING>	-	Square bracket enclosed list of node features required by job if specified. (For example: [fast][ethernet])
<b>System Queue Time</b>	24	<INTEGER>	0	Epoch time when job met all fairness policies.
<b>Tasks Allocated</b>	25	<INTEGER>	<TASKS REQUESTED>	Number of tasks actually allocated to job. <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin-top: 10px;">  In most cases, this field is identical to field #7, Tasks Requested.                 </div>
<b>Required Tasks Per Node</b>	26	<INTEGER>	-1	Number of Tasks Per Node required by job or '-1' if no requirement specified.

Field Name	Field Index	Data Format	Default Value	Details
<b>QoS</b>	27	<STRING> [:<STRING>]	-	QoS requested/assigned using the format <QOS_REQUESTED>[:<QOS_DELIVERED>]. (For example: hipriority:bottomfeeder)
<b>JobFlags</b>	28	<STRING> [:<STRING>]...	-	Square bracket delimited list of job attributes. (For example: [BACKFILL][PREEMPTEE])
<b>Account Name</b>	29	<STRING>	-	Name of account associated with job if specified.
<b>Executable</b>	30	<STRING>	-	Name of job executable if specified.
<b>Resource Manager Extension String</b>	31	<STRING>	-	Resource manager specific list of job attributes if specified. See the <a href="#">Resource Manager Extension Overview</a> for more information.
<b>Bypass Count</b>	32	<INTEGER>	-1	Number of times job was bypassed by lower priority jobs via backfill or '-1' if not specified.
<b>ProcSeconds Utilized</b>	33	<DOUBLE>	0	Number of processor seconds actually used by job.
<b>Partition Name</b>	34	<STRING>	[DEFAULT]	Name of partition in which job ran.
<b>Dedicated Processors per Task</b>	35	<INTEGER>	1	Number of processors required per task.
<b>Dedicated Memory per Task</b>	36	<INTEGER>	0	Amount of RAM (in MB) required per task.
<b>Dedicated Disk per Task</b>	37	<INTEGER>	0	Amount of local disk (in MB) required per task.
<b>Dedicated Swap per Task</b>	38	<INTEGER>	0	Amount of virtual memory (in MB) required per task.

Field Name	Field Index	Data Format	Default Value	Details
<b>Start Date</b>	39	<INTEGER>	0	Epoch time indicating earliest time job can start.
<b>End Date</b>	40	<INTEGER>	0	Epoch time indicating latest time by which job must complete.
<b>Allocated Host List</b>	41	<hostname> [, <hostname> ]...	-	Comma delimited list of hosts allocated to job. (For example: node001,node004)
<b>Resource Manager Name</b>	42	<STRING>	-	Name of resource manager if specified.
<b>Required Host List</b>	43	<hostname> [, <hostname> ]...	-	List of hosts required by job. (If the job's taskcount is greater than the specified number of hosts, the scheduler must use these nodes in addition to others; if the job's taskcount is less than the specified number of hosts, the scheduler must select needed hosts from this list.)
<b>Reservation</b>	44	<STRING>	-	Name of reservation required by job if specified.
<b>Application Simulator Data</b>	45	<STRING> [: <STRING>]	-	Name of application simulator module and associated configuration data. (For example: HSM:IN- N=infile.txt:140000;OUT= outfile.txt:500000)
<b>Set Description</b>	46	<STRING>: <STRING> [: <STRING>]	-	Set constraints required by node in the form <SetConstraint>:<SetType>[:<SetList>] where SetConstraint is one of ONEOF, FIRSTOF, or ANYOF, SetType is one of PROCSPEED, FEATURE, or NETWORK, and SetList is an optional colon delimited list of allowed set attributes. (For example: ONEOF:PROCSPEED:350:450:500)

Field Name	Field Index	Data Format	Default Value	Details
<b>Job Message</b>	47	<STRING>	-	Job messages including resource manager, scheduler, and administrator messages if specified.
<b>Job Cost</b>	48	<DOUBLE>	0.0	Cost of executing job incorporating resource consumption metric, resource quantity consumed, and credential, allocated resource, and delivered QoS charge rates.
<b>History</b>	49	<STRING>	-	List of job events impacting resource allocation (XML).  <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF;">  History information is only reported in Moab 5.1.0 and higher.                 </div>
<b>Utilization</b>	50	Comma-delimited list of one or more of the following: <ATTR>= <VALUE> pairs where <VALUE> is a double and <ATTR> is one of the following: network (in MB transferred), license (in license-seconds), storage (in MB-seconds stored), or gmetric: <TYPE>.	-	Cumulative resources used over life of job.
<b>Estimate Data</b>	51	<STRING>	-	List of job estimate usage.
<b>Completion Code</b>	52	<INTEGER>	-	Job exit status/completion code.

Field Name	Field Index	Data Format	Default Value	Details
<b>Extended Memory Load Information</b>	53	<STRING>	-	<b>Deprecated.</b> Extended memory usage statistics (max, mem, avg, and so forth).
<b>Extended CPU Load Information</b>	54	<STRING>	-	Extended CPU usage statistics (max, mem, avg, and so forth).
<b>Generic Metric Averages</b>	55	<STRING>	-1	Generic metric averages.
<b>Effective Queue Duration</b>	56	<INTEGER>	-1	The amount of time, in seconds, that the job was eligible for scheduling.
<b>Job Submission Arguments</b>	57	<STRING>	-	The job's submit arguments and script. This field is enabled by setting <a href="#">STOREJOBSSUBMISSION</a> to <i>TRUE</i> .

**i** If no applicable value is specified, the exact string – should be entered.

**i** Fields that contain a description string such as Job Message use a packed string format. The packed string format replaces white space characters such as spaces and carriage returns with a hex character representation. For example a blank space is represented as \20. Since fields in the event record are space delimited, this preserves the correct order and spacing of fields in the record.

### Sample Workload Trace

```
13:21:05 110244355 job 1413 JOBEND 20 20 josh staff 86400 Removed [batch:1] 887343658
889585185 \
889585185 889585411 ethernet R6000 AIX53 >= 256 >= 0 - 889584538 20 0 0 2 0 test.cmd \
1001 6 678.08 0 1 0 0 0 0 0 - 0 - - - - - 0.0 - - - 0 - -
```

### Creating New Workload Simulation Traces

Because workload [event records](#) and simulation workload traces use the same format, these event records can be used as a starting point for generating a new simulation trace. In the Moab simple case, an event record or collection of event records can be used directly as the value for the [SIMWORKLOADTRACEFILE](#) as in the following example:

1. 

```
# collect all job records for July
> cat /opt/moab/stats/events.*July*2012 | grep JOBEND > /opt/moab/DecJobs.txt
```

```
2. edit moab.cfg for use job records
> vi /opt/moab/etc/moab.cfg
(add 'SIMWORKLOADTRACEFILE /opt/moab/DecJobs.txt')
```

**i** In the preceding example, all non-*JOBEND* events were filtered out. This step is not required but only *JOBEND* events are used in a simulation; other events are ignored by Moab.

### Modifying Existing Job Event Records

When creating a new simulation workload, it is often valuable to start with workload traces representing a well-known or even local workload. These traces preserve distribution information about job submission times, durations, processor count, users, groups, projects, special resource requests, and numerous other factors that effectively represent an industry, user base, or organization.

When modifying records, a field or combination of fields can be altered, new jobs inserted, or certain jobs filtered out.

**i** Because job event records are used for multiple purposes, some of the fields are valuable for statistics or auditing purposes but are ignored in simulations. For the most part, fields representing resource utilization information are ignored while fields representing resource requests are not.

### Modifying Time Distribution Factors of a Workload Trace

In some cases, simulations focus on determining the effects of changing the quantities or types of jobs or on changing policies or job ownership to see changes to system performance and resource utilization. However, other times simulations tend to focus on response-time metrics as job submission and job duration aspects of the workload are modified. Which time-based fields are important to modify depend on the simulation purpose and the setting of the [JOBSUBMISSIONPOLICY](#) parameter.

JOBSUBMISSIONPOLICY Value	Critical Time Based Fields
<b>NORMAL</b>	WallClock Limit Submission Time StartTime Completion Time
<b>CONSTANTJOBDEPTH</b> <b>CONSTANTPSDEPTH</b>	WallClock Limit StartTime Completion Time

**i** Dispatch Time should always be identical to Start Time.

**i** In all cases, the difference of 'Completion Time - Start Time' is used to determine actual job run time.

**i** System Queue Time and Proc-Seconds Utilized are only used for statistics gathering purposes and will not alter the behavior of the simulation.

**i** In all cases, relative time values are important, i.e., Start Time must be greater than or equal to Submission Time and less than Completion Time.

### Creating Workload Traces From Scratch

There is nothing which prevents a completely new workload trace from being created from scratch. To do this, simply create a file with fields matching the format described in the [Workload Event Record Format](#) section.

### Reservation Records/Traces

All reservation events provide reservation data in a standard format as described in the following table:

Field Name	Field Index	Data Format	Default Value	Details
<b>Event Time (Human)</b>	0	[HH:MM:SS]	-	Specifies time event occurred.
<b>Event Time (Epoch)</b>	1	<epochtime>	-	Specifies time event occurred.
<b>Object Type</b>	2	rsv	-	Specifies record object type.
<b>Object ID</b>	3	<STRING>	-	Unique object identifier.
<b>Object Event</b>	4	one of <i>rsvcreate</i> , <i>rsvstart</i> , <i>rsvmodify</i> , <i>rsvfail</i> or <i>rsvend</i>	-	Specifies record event type.
<b>Creation Time</b>	5	<EPOCHTIME>	-	Specifies epoch time of reservation start date.
<b>Start Time</b>	6	<EPOCHTIME>	-	Specifies epoch time of reservation start date.
<b>End Time</b>	7	<EPOCHTIME>	-	Specifies epoch time of reservation end date.

Field Name	Field Index	Data Format	Default Value	Details
<b>Tasks Allocated</b>	8	<INTEGER>	-	Specifies number of tasks allocated to reservation at event time.
<b>Nodes Allocated</b>	9	<INTEGER>	-	Specifies number of nodes allocated to reservation at event time.
<b>Total Active Proc-Seconds</b>	10	<INTEGER>	-	Specifies proc-seconds reserved resources were dedicated to one or more job at event time.
<b>Total Proc-Seconds</b>	11	<INTEGER>	-	Specifies proc-seconds resources were reserved at event time.
<b>Hostlist</b>	12	<comma-delimited list of hostnames>	-	Specifies list of hosts reserved at event time.
<b>Owner</b>	13	<STRING>	-	Specifies reservation <a href="#">ownership</a> credentials.
<b>ACL</b>	14	<STRING>	-	Specifies reservation <a href="#">access control list</a> .
<b>Comment</b>	15	<STRING>	-	Specifies general human readable event message.
<b>Command Line</b>	16	<STRING>	-	Displays the command line arguments used to create the reservation (only shows on the rsvcreate event).

## Recording Job Events

Job events occur when a job undergoes a definitive change in state. Job events include submission, starting, cancellation, migration, and completion. Some site administrators do not want to use an external accounting system and use these logged events to determine their clusters' accounting statistics. Moab can be configured to record these events in the appropriate event file found in the Moab `stats/` directory. To enable job event recording for both local and remotely staged jobs, use the [RECORDEVENTLIST](#) parameter. For example:

```
RECORDEVENTLIST JOBCANCEL, JOBCOMPLETE, JOBSTART, JOBSUBMIT
...
```

*This configuration records an event each time both remote and/or local jobs are canceled, run to completion, started, or submitted. The [Event Logs](#) section details the format of these records.*

## Related topics

- [Event Logging Overview](#)
- [SIMWORKLOADTRACEFILE](#)

## Interactive Simulation Tutorial

This of this section provides an interactive tutorial to demonstrate the basics of the simulator's capacities in Moab. It is an example of what you can do once you have set up simulation. The commands to issue are formatted as follows: > `showq` along with the expected output.

The following commands are used:

- [showq \[-r\] \[-i\]](#)
- [showstats \[-g\] \[-u\] \[-v\]](#)
- [mschedctl -l](#)
- [mschedctl \[{-s|-S} \[I\]\] \[-k\]](#)
- [checkjob](#)
- [mschedctl -m](#)
- [mdiag -n](#)
- [showres \[-n jobid\]](#)
- [setres](#)

## To run through the simulation mode tutorial

1. Run moab.

```
> moab&
```

2. Check the status of the queue (see [Checking the Queue Status](#)).
3. Check the status of the job. If any jobs are not running, find the problem (see [Determining Why Jobs Are Not Running](#)).
4. Advance and check the status of Moab iterations and time in simulation mode (see [Controlling Iterations](#)).
5. View and manage reservations and their nodes and jobs (see [Managing Reservations Applying to the Queue](#)).
6. Verify that the Moab simulation is scheduling fairly (see [Verifying Fair Scheduling](#)).
7. Take down the entire system for maintenance (see [Taking the System Down for Maintenance](#)).

### Checking the Queue Status

Verify that Moab is running by executing `showq`:

```

> showq
active jobs-----
JOBNAME          USERNAME      STATE  PROC   REMAINING          STARTTIME
fr8n01.187.0      570          Running 20  1:00:00:00  Mon Feb 16 11:54:03
fr8n01.189.0      570          Running 20  1:00:00:00  Mon Feb 16 11:54:03
fr8n01.190.0      570          Running 20  1:00:00:00  Mon Feb 16 11:54:03
fr8n01.191.0      570          Running 20  1:00:00:00  Mon Feb 16 11:54:03
fr8n01.276.0      550          Running 20  1:00:00:00  Mon Feb 16 11:54:03
frln04.369.0      550          Running 20  1:00:00:00  Mon Feb 16 11:54:03
frln04.487.0      550          Running 20  1:00:00:00  Mon Feb 16 11:54:03
  7 active jobs    140 of 196 Processors Active (71.43%)
eligible jobs-----
JOBNAME          USERNAME      STATE  PROC   WCLIMIT          QUEUETIME
frln04.362.0      550          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.363.0      550          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.365.0      550          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.366.0      550          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.501.0      570          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.580.0      570          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.597.0      570          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.598.0      570          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
frln04.602.0      570          Idle   20   1:00:00:00  Mon Feb 16 11:53:33
  9 eligible jobs
blocked jobs-----
JOBNAME          USERNAME      STATE  PROC   WCLIMIT          QUEUETIME
0 blocked jobs
Total jobs: 16

```

*Out of the thousands of jobs in the workload trace, only 16 jobs are either active or eligible because of the default settings of the [SIMINITIALQUEUEDEPTH](#) parameter. Sixteen jobs are put in the idle queue, seven of which immediately run. Issuing the command `showq -r` allows a more detailed look at the active (or running) jobs. The output is sorted by job completion time and indicates that the first job will complete in one day (1:00:00:00).*

## Determining Why Jobs Are Not Running

While `showq` details information about the queues, scheduler statistics may be viewed using the [showstats](#) command. The field Current Active/Total Procs shows current system utilization, for example.

```

> showstats
moab active for      00:00:30  stats initialized on Mon Feb 16 11:53:33
Eligible/Idle Jobs:          9/9          (100.000%)
Active Jobs:                  0
Successful/Completed Jobs:    0/0          (0.000%)
Avg/Max QTime (Hours):        0.00/0.00
Avg/Max XFactor:              0.00/0.00
Dedicated/Total ProcHours:    1.17/1.63    (71.429%)

Current Active/Total Procs:      140/196      (71.429%)

Avg WallClock Accuracy:      N/A
Avg Job Proc Efficiency:      N/A
Est/Avg Backlog (Hours):      N/A / N/A

```

You might be wondering why there are only 140 of 196 Processors Active (as shown with `showq`) when the first job (`frln04.362.0`) in the queue only requires 20 processors. We will use the [checkjob](#) command, which reports detailed job state information and diagnostic output for a particular job to determine why it is not running:

```

> checkjob frln04.362.0
job frln04.362.0
State: Idle
...
Network: hps_user  Memory >= 256M  Disk >= 0  Swap >= 0
...
Job Eligibility Analysis -----
job cannot run in partition DEFAULT (idle procs do not meet requirements : 8 of 20
procs found)
idle procs: 56  feasible procs: 8
Rejection Reasons: [Memory : 48][State : 140]

```

*checkjob not only tells us the job's wallclock limit and the number of requested nodes (they're in the ellipsis) but explains why the job was rejected from running. The Job Eligibility Analysis tells us that 48 of the processors rejected this job due to memory limitations and that another 140 processors rejected it because of their state (that is, they're running other jobs). Notice the >= 256 M(B) memory requirement.*

If you run `checkjob` with the ID of a running job, it would also tell us exactly which nodes have been allocated to this job. There is additional information that the [checkjob](#) command page describes in more detail.

### Controlling Iterations

Advancing the simulator an iteration, the following happens:

```

> mschedctl -S
scheduling will stop in 00:00:30 at iteration 1

```

The scheduler control command, [mschedctl](#), controls various aspects of scheduling behavior. It can be used to manage scheduling activity, kill the scheduler, and create resource trace files. The `-S` argument indicates that the scheduler run for a single iteration and stop. Specifying a number, *n*, after `-S` causes the simulator to advance *n* steps. You can determine what iteration you are currently on using `showstats -v`.

```

> showstats -v
current scheduler time: Mon Feb 16 11:54:03 1998 (887655243)
moab active for      00:01:00  stats initialized on Mon Feb 16 11:53:33
statistics for iteration      1  scheduler started on Wed Dec 31 17:00:00
...

```

*The line that starts with statistics for iteration <X> specifies the iteration you are currently on. Each iteration advances the simulator **RMPOLLINTERVAL** seconds. By default, **RMPOLLINTERVAL** is set to 30 seconds.*

To see what **RMPOLLINTERVAL** is set to, use the [showconfig](#) command:

```

> showconfig | grep RMPOLLINTERVAL
RMPOLLINTERVAL      30,30

```

The `showq -r` command can be used to display the running (active) jobs to see what happened in the last iteration:

```

> showq -r
active jobs-----
JOBID          S PAR  EFFIC  XFACTOR  Q      USER   GROUP   MHOST  PROCS
REMAINING      STARTTIME
fr8n01.804.0    R  1  -----    1.0  -      529    519    fr9n16   5
00:05:00 Mon Feb 16 11:54:03
fr8n01.187.0    R  1  -----    1.0  -      570    519    fr7n15  20
1:00:00:00 Mon Feb 16 11:54:03
...
fr8n01.960.0    R  1  -----    1.0  -      588    519    fr9n11  32
1:00:00:00 Mon Feb 16 11:54:03
  9 active jobs      177 of 196 Processors Active (90.31%)
Total jobs:  9

```

*Notice that two new jobs started (without waiting in the eligible queue). Also notice that job fr8n01.187.0, along with the rest that are summarized in the ellipsis, did NOT advance its REMAINING or STARTTIME.*

The simulator needs one iteration to do a sanity check. Setting the parameter **STOPITERATION** to 1 causes Moab to stop after the first scheduling iteration and wait for further instructions.

The `showq -i` command displays the idle (eligible) jobs.

```

> showq -i
eligible jobs-----
JOBID          PRIORITY  XFACTOR  Q      USER   GROUP  PROCS   WCLIMIT
CLASS          SYSTEMQUEUEUETIME
frln04.362.0*  1         1.0  -      550    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.363.0  1         1.0  -      550    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.365.0  1         1.0  -      550    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.366.0  1         1.0  -      550    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.501.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.580.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.597.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.598.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.602.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:53:33
frln04.743.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:54:03
frln04.744.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:54:03
frln04.746.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:54:03
frln04.747.0  1         1.0  -      570    519    20  1:00:00:00
batch Mon Feb 16 11:54:03
fr8n01.388.0  1         1.0  -      550    519    20  1:00:00:00
batch Mon Feb 16 11:54:03
14 eligible jobs
Total jobs: 14

```

*Notice how none of the eligible jobs are requesting 19 or fewer jobs (the number of idle processors). Also notice the \* after the job id frln04.362.0. This means that this job now has a reservation.*

## Managing Reservations Applying to the Queue

The `showres` command shows all reservations currently on the system.

```
> showres
ReservationID      Type S      Start      End      Duration  N/P      StartTime
fr8n01.187.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr8n01.189.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr8n01.190.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr8n01.191.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr8n01.276.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr1n04.362.0      Job I      1:00:00:00 2:00:00:00 1:00:00:00 20/20    Tue Feb 17
11:54:03
fr1n04.369.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr1n04.487.0      Job R      00:00:00   1:00:00:00 1:00:00:00 20/20    Mon Feb 16
11:54:03
fr8n01.804.0      Job R      00:00:00   00:05:00   00:05:00   5/5      Mon Feb 16
11:54:03
fr8n01.960.0      Job R      00:00:00   1:00:00:00 1:00:00:00 32/32    Mon Feb 16
11:54:03
10 reservations located
```

*Here, the S column is the job's state (R = running, I = idle). All the active jobs have a reservation along with idle job fr1n04.362.0. This reservation was actually created by the backfill scheduler for the highest priority idle job as a way to prevent starvation while lower priority jobs were being backfilled (The [backfill documentation](#) describes the mechanics of the backfill scheduling more fully).*

To display information about the nodes that job fr1n04.362.0 has reserved, use `showres -n <JOBID>`.

```
> showres -n fr1n04.362.0
reservations on Mon Feb 16 11:54:03
NodeName          Type      ReservationID  JobState Task      Start
Duration  StartTime
fr5n09          Job      fr1n04.362.0  Idle    1  1:00:00:00
1:00:00:00  Tue Feb 17 11:54:03
...
fr7n15          Job      fr1n04.362.0  Idle    1  1:00:00:00
1:00:00:00  Tue Feb 17 11:54:03
20 nodes reserved
```

Now advance the simulator an iteration to allow some jobs to actually run.

```
> mschedctl -S
scheduling will stop in 00:00:30 at iteration 2
```

Next, check the queues to see what happened.

```

> showq
active jobs-----
JOBNAME          USERNAME      STATE  PROC   REMAINING          STARTTIME
fr8n01.804.0      529          Running  5     00:04:30  Mon Feb 16 11:54:03
fr8n01.187.0      570          Running  20    23:59:30  Mon Feb 16 11:54:03
...
    9 active jobs      177 of 196 Processors Active (90.31%)
eligible jobs-----
JOBNAME          USERNAME      STATE  PROC   WCLIMIT          QUEUETIME
...
fr8n01.963.0      586          Idle    32     9:00:00  Mon Feb 16 11:54:33
fr8n01.1016.0    570          Idle    20    1:00:00:00  Mon Feb 16 11:54:33
16 eligible jobs
...

```

Two new jobs, fr8n01.963.0 and fr8n01.1016.0, are in the eligible queue. Also, note that the first job will now complete in 4 minutes 30 seconds rather than 5 minutes because we have just advanced now by 30 seconds, one **RMPOLLINTERVAL**. It is important to note that when the simulated jobs were created, both the job's wallclock limit and its actual run time were recorded. The wallclock limit is specified by the user indicating their best estimate of an upper bound on how long the job will run. The *run time* is how long the job actually ran before completing and releasing its allocated resources. For example, a job with a wallclock limit of 1 hour will be given the needed resources for up to an hour but may complete in only 20 minutes.

Stop the simulation at iteration 6.

```

> mschedctl -s 6I
scheduling will stop in 00:03:00 at iteration 6

```

*The -s 6I argument indicates that the scheduler will stop at iteration 6 and will (I)gnore user input until it gets there. This prevents the possibility of obtaining showq output from iteration 5 rather than iteration 6.*

```

> showq
active jobs-----
JOBNAME          USERNAME      STATE  PROC   REMAINING          STARTTIME
fr8n01.804.0      529          Running  5     00:02:30  Mon Feb 16 11:54:03
...
fr1n04.501.0      570          Running  20    1:00:00:00  Mon Feb 16 11:56:33
fr8n01.388.0      550          Running  20    1:00:00:00  Mon Feb 16 11:56:33
    9 active jobs      177 of 196 Processors Active (90.31%)
...
    14 eligible jobs
...

```

*Job fr8n01.804.0 is still 2 minutes 30 seconds away from completing as expected, but notice that both jobs fr8n01.189.0 and fr8n01.191.0 have completed early. Although they had almost 24 hours remaining of wallclock limit, they terminated. In reality, they probably failed on the real world system where the trace file was being created. Their completion freed up 40 processors which the scheduler was able to immediately use by starting several more jobs.*

Note the system statistics:

```

> showstats
...
Successful/Completed Jobs:          0/2          (0.000%)
...
Avg WallClock Accuracy:             0.150%
Avg Job Proc Efficiency:            100.000%
Est/Avg Backlog (Hours):            0.00/3652178.74

```

A few more fields are filled in now that some jobs have completed providing information on which to generate statistics.

Decrease the default [LOGLEVEL](#) with `mschedctl -m` to avoid unnecessary logging, and speed up the simulation.

```
> mschedctl -m LOGLEVEL 0
INFO: parameter modified
```

*You can use `mschedctl -m` to immediately change the value of any parameter. The change is only made to the currently running Moab server and is not propagated to the configuration file. Changes can also be made by modifying the configuration file and restarting the scheduler.*

Stop at iteration 580 and pull up the scheduler's statistics.

```
> mschedctl -s 580I; showq
scheduling will stop in 4:47:00 at iteration 580
...
  11 active jobs      156 of 196 Processors Active (79.59%)
eligible jobs-----
JOBNAME              USERNAME      STATE  PROC    WCLIMIT      QUEUETIME
fr8n01.963.0         586          Idle   32     9:00:00  Mon Feb 16 11:54:33
fr8n01.1075.0        560          Idle   32    23:56:00  Mon Feb 16 11:58:33
fr8n01.1076.0        560          Idle   16    23:56:00  Mon Feb 16 11:59:33
fr1n04.1953.0        520          Idle   46     7:45:00  Mon Feb 16 12:03:03
...
16 eligible jobs
...
```

*You may note that `showq` hangs a while as the scheduler simulates up to iteration 580. The output shows that currently only 156 of the 196 nodes are busy, yet at first glance 3 jobs, `fr8n01.963.0`, `fr8n01.1075.0`, and `fr8n01.1076.0` appear to be ready to run.*

```

> checkjob fr8n01.963.0; checkjob fr8n01.1075.0; checkjob fr8n01.1076.0
job fr8n01.963.0
...
Network: hps_user  Memory >= 256M  Disk >= 0  Swap >= 0
...
Job Eligibility Analysis -----
job cannot run in partition DEFAULT (idle procs do not meet requirements : 20 of 32
procs found)
idle procs: 40  feasible procs: 20
Rejection Reasons: [Memory : 20][State : 156]

job fr8n01.1075.0
...
Network: hps_user  Memory >= 256M  Disk >= 0  Swap >= 0
...
job cannot run in partition DEFAULT (idle procs do not meet requirements : 0 of 32
procs found)
idle procs: 40  feasible procs: 0
Rejection Reasons: [Memory : 20][State : 156][ReserveTime : 20]

job fr8n01.1076.0
...
Network: hps_user  Memory >= 256M  Disk >= 0  Swap >= 0
...
job cannot run in partition DEFAULT (idle procs do not meet requirements : 0 of 16
procs found)
idle procs: 40  feasible procs: 0
Rejection Reasons: [Memory : 20][State : 156][ReserveTime : 20]

```

*The checkjob command reveals that job fr8n01.963.0 only found 20 of 32 processors. The remaining 20 idle processors could not be used because the configured memory on the node did not meet the jobs requirements. The other jobs cannot find enough nodes because of ReserveTime. This indicates that the processors are idle, but that they have a reservation in place that will start before the job being checked could complete.*

Verify that the idle nodes do not have enough memory configured and they are already reserved with the [mdiag -n](#) command, which provides detailed information about the state of nodes Moab is currently tracking. The [mdiag](#) command can be used with various flags to obtain detailed information about [accounts](#), [blocked jobs](#), [fair share](#), [groups](#), [jobs](#), [nodes](#), [QoS](#), [reservations](#), the [resource manager](#), and [users](#). The command also performs a number of sanity checks on the data provided and will present warning messages if discrepancies are detected.

```

> mdiag -n -v | grep -e Name -e Idle
Name      State  Procs Memory      Disk      Swap      Speed  Opsys  Arch  Par
Load Rsv  ...
fr10n09  Idle  1:1  256:256    9780:9780  411488:411488  1.00  AIX43  R6000 DEF
0.00 001  .
fr10n11  Idle  1:1  256:256    8772:8772  425280:425280  1.00  AIX43  R6000 DEF
0.00 001  .
fr10n13  Idle  1:1  256:256    9272:9272  441124:441124  1.00  AIX43  R6000 DEF
0.00 001  .
fr10n15  Idle  1:1  256:256    8652:8652  440776:440776  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n01  Idle  1:1  256:256    7668:7668  438624:438624  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n03  Idle  1:1  256:256    9548:9548  424584:424584  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n05  Idle  1:1  256:256   11608:11608 454476:454476  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n07  Idle  1:1  256:256    9008:9008  425292:425292  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n09  Idle  1:1  256:256    8588:8588  424684:424684  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n11  Idle  1:1  256:256    9632:9632  424936:424936  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n13  Idle  1:1  256:256    9524:9524  425432:425432  1.00  AIX43  R6000 DEF
0.00 001  .
fr11n15  Idle  1:1  256:256    9388:9388  425728:425728  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n01  Idle  1:1  256:256    6848:6848  424260:424260  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n03  Idle  1:1  256:256    9752:9752  424192:424192  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n05  Idle  1:1  256:256    9920:9920  434088:434088  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n07  Idle  1:1  256:256    2196:2196  434224:434224  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n09  Idle  1:1  256:256    9368:9368  434568:434568  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n11  Idle  1:1  256:256    9880:9880  434172:434172  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n13  Idle  1:1  256:256    9760:9760  433952:433952  1.00  AIX43  R6000 DEF
0.00 001  .
fr14n15  Idle  1:1  256:256   25000:25000 434044:434044  1.00  AIX43  R6000 DEF
0.00 001  .
fr17n05  Idle  1:1  128:128    10016:10016 182720:182720  1.00  AIX43  R6000 DEF
0.00 000  .
...
Total Nodes: 196 (Active: 156 Idle: 40 Down: 0)

```

*The grep gets the command header and the idle nodes listed. All the idle nodes with 256 MB of memory installed already have a reservation. (See the Rsv column.) The rest of the idle nodes only have 128 MB of memory.*

```

> checknode fr10n09
node fr10n09
State:      Idle (in current state for 4:21:00)
Configured Resources: PROCS: 1 MEM: 256M SWAP: 401G DISK: 9780M
Utilized Resources: [NONE]
Dedicated Resources: [NONE]
..
Total Time: 4:50:00 Up: 4:50:00 (100.00%) Active: 00:34:30 (11.90%)
Reservations:
Job 'fr8n01.963.0' (x1) 3:25:00 -> 12:25:00 (9:00:00)

```

*Using `checknode` revealed that Job `fr8n01.963.0` has the reservation.*

### Moving ahead:

```

> mschedctl -S 500I;showstats -v
scheduling will stop in 4:10:00 at iteration 1080
...
Eligible/Idle Jobs:          16/16          (100.000%)
Active Jobs:                 11
Successful/Completed Jobs:   2/25          (8.000%)
Preempt Jobs:                0
Avg/Max QTime (Hours):      0.00/0.00
Avg/Max XFactor:            0.00/1.04
Avg/Max Bypass:             0.00/13.00
Dedicated/Total ProcHours:   1545.44/1765.63 (87.529%)
Preempt/Dedicated ProcHours: 0.00/1545.44 (0.000%)
Current Active/Total Procs:  156/196        (79.592%)
Avg WallClock Accuracy:     9.960%
Avg Job Proc Efficiency:     100.000%
Min System Utilization:     79.592% (on iteration 33)
Est/Avg Backlog (Hours):    0.00/20289.84

```

*We now know that the scheduler is scheduling efficiently. So far, system utilization as reported by `showstats -v` looks very good.*

### Verifying Fair Scheduling

An important and subjective question is whether the scheduler is scheduling fairly. Look at the user and group statistics to see if there are any glaring problems.

```
> showstats -u
statistics initialized Wed Dec 31 17:00:00
-----|----- Active -----|----- Completed -----
-----|
user      Jobs Procs ProcHours Jobs   %   PHReq   %   PHDed   %   FSTgt  AvgXF
MaxXF    AvgQH  Effic  WCAcc
520      1    46    172.88   1   0.00  356.5   0.00  541.3   0.00  -----  1.04
0.00    0.35 100.00 100.00
550      1    20    301.83   7   0.00 3360.0   0.00  283.7   0.00  -----  0.03
0.00    0.06 100.00   3.17
524      1    32    239.73   ---  ---  ---  ---  272.3   0.00  -----  ---
----- 100.00 -----
570      1    20    301.00  14   0.00 6720.0   0.00  199.5   0.00  -----  0.01
0.00    0.20 100.00   0.34
588      0     0     0.00    1   0.00  768.0   0.00  159.7   0.00  -----  0.21
0.00    0.00 100.00  20.80
578      6     6    146.82   ---  ---  ---  ---  53.2   0.00  -----  ---
----- 100.00 -----
586      1    32    265.07   ---  ---  ---  ---  22.9   0.00  -----  ---
----- 100.00 -----
517      0     0     0.00    1   0.00  432.0   0.00   4.8   0.00  -----  0.02
0.00    0.12 100.00   1.10
529      0     0     0.00    1   0.00   0.4   0.00   1.3   0.00  -----  1.00
0.00    0.00 100.00 100.00
```

```
> showstats -g
statistics initialized Wed Dec 31 17:00:00
-----|----- Active -----|----- Completed -----
-----|
group     Jobs Procs ProcHours Jobs   %   PHReq   %   PHDed   %   FSTgt  AvgXF
MaxXF    AvgQH  Effic  WCAcc
503      1    32    239.73   1   0.00  432.0   0.00  277.1   0.00  -----  0.02
0.00    0.12 100.00   1.10
501      1    32    265.07   ---  ---  ---  ---  22.9   0.00  -----  ---
----- 100.00 -----
519      9    92    922.54  24   0.00 11204.9  0.00 1238.6  0.00  -----  0.11
0.00    0.15 100.00  10.33
```

### Taking the System Down for Maintenance

Suppose you need to now take down the entire system for maintenance on Thursday from 2:00 to 8:00 a.m. To do this, create a reservation with [mrsvctl -c](#).

```
> mrsvctl -c -t ALL -s 2:00_02/17 -d 6:00:00
```

Shut down the scheduler.

```
> mschedctl -k
moab will be shutdown immediately
```

# Green computing

## Green computing overview

SearchDataCenter.com defines green computing as the environmentally responsible use of computers and related resources. Such practices include the implementation of energy-efficient central processing units (CPUs), servers, and peripherals as well as reduced resource consumption and proper disposal of electronic waste (e-waste).

The Moab HPC Suites, both Basic Edition and Enterprise Edition, contain power management features that give a Moab administrator the ability to implement policies that can conserve energy and save on operational costs, often without affecting an HPC system's performance with regard to job execution times.

Effective power management means managing power or energy consumption while a compute node is actively running jobs, and when a compute node is idle. Both scenarios require different tools and policies.

- Active compute node power management is mainly performed through control of the clock frequency of the processor(s) on a compute node while a job is executing. Decreasing the clock frequency can reduce energy usage.
- Idle compute node power management is mainly performed by placing a compute node into different low-power states, such as standby and suspend, or no-power states, such as hibernate and shutdown.

The table below identifies the Moab power management features and/or methods available for the different Moab HPC Suite editions.

Feature or Method	Moab HPC Suite Edition	
	Basic	Enterprise
CPU Clock Frequency Control		
• Moab Job Submission Option	X	X
• TORQUE Job Submission Option	X	X
• Moab Job Template Option	X	X
Manual Power Management		
• Moab-based on and off states	-	X
• TORQUE-based low-power and no-power states	X	X

Feature or Method	Moab HPC Suite Edition	
	Basic	Enterprise
Automated Power Management and Green Policies		
<ul style="list-style-type: none"> <li>• Moab-only global-level policies and power management for <code>on</code> and <code>off</code> states</li> <li>• Moab/Moab Web Services-based global, partition, and node-level policies and power management for low-power and no-power states</li> <li>• Green Idle Node Pool Management Policies</li> </ul>	-	X
	-	X
	-	X

## Moab Power Management Methods

Moab supports two separate and mutually-exclusive methods for managing the power state of compute nodes, which affects energy consumption. The first method, introduced in Moab 7.2, allows an administrator to manually power on and power off compute nodes and to create a global set of green policies that automatically perform these two functions based on specific conditions involving idle compute nodes. The second method, introduced in Moab 8.0 and TORQUE 5.0, give an administrator additional power states besides `on` and `off` and offer finer control of green policies at the global, partition, and node levels. Before delving into the theory of operation of these two separate methods, an administrator must understand how Moab views power management regardless of which method is used.

### Moab View of Power Management

Moab is not aware of the actual power state of nodes. From Moab's perspective, nodes are only on or off. If Moab needs a node that is off, it issues a power-on job prior to scheduling the incoming job.

In addition, in order to schedule a job to a compute node, Moab requires the compute node's workload resource manager, which in our example is TORQUE, to report the compute node's state is `idle`. When the compute node's binary power state indicates `on` and the RM indicates the compute node's state is `idle`, Moab will schedule jobs to the compute node. Any value other than `idle` for the node's state and Moab will not schedule a job to the node. If the power state is `off`, Moab issues a power-on job as a dependency to the regular job.

Moab performs compute node power management entirely through power management resource managers, or Power RMs. Each of the two power management methods mentioned above has its own Power RM implementation. The older Moab-only method uses Python-based scripts to implement a power RM while the newer Moab+Moab Web Services (MWS)-based method uses a Java-based MWS RM power management plug-in that runs much simpler Python-based scripts.

These Power RMs perform all power-related management and monitoring, meaning power state control and power state query, respectively, and only report back to Moab whether a compute node is in a state in which it can run jobs (`on`) or not (`off`). All actual power state-aware control and management is performed by the power RMs.

## Moab Power RMs

Adaptive Computing provides two power management methods to handle different site scenarios; mainly for site-specific security policies. The older method handles sites with a security policy that does not permit web service-based services, which can be an attack vector, or sites that do not want to run an MWS service.

The newer method uses the MWS RM plug-in feature, which allows an administrator to instantiate a separate RM power management plug-in instance for different partitions, or different compute nodes for situations where different compute node hardware requires the use of different power management commands run from Python scripts.

## Power Management Scripts

Each power management method, old or new, employs at some point a script that allows the administrator to customize power management for a site, which may be required because the working reference scripts provided by Adaptive Computing (based on OpenIPMI tools) do not use the power management commands specific to the site's vendor-provided hardware.

## Moab System Jobs

Moab performs power management functions through a mechanism known as system jobs. A Moab system job is a special, separately scheduled job that performs some Moab system function (e.g., power management, data-staging) that Moab executes on the Moab head node and not on a compute node. This allows Moab to apply policies such as a job wallclock estimate, etc, to system-related functions, which can aid error recovery procedures, etc.

System jobs perform internal Moab-related functions on Moab's behalf, are nearly always script-based, and usually require some customization by the Moab administrator in order to perform the needed function for the HPC system site. For example, the administrator may have to modify power management scripts so they use a site's hardware vendor-specific power management commands to effect power state changes in compute nodes.

To create a system job, Moab internally submits an administrator-defined script, with a path typically specified as a Moab `*URL` parameter, to itself, which it flags as a system job. Moab schedules the job and because it is flagged as a system job, executes the script on the head node. Moab submits a system job whenever it needs to send a power on or off command to a Power RM. Administrators can easily recognize queued and running power management system jobs in the `showq` command output as their job id has the format `id.poweron` and `id.poweroff`, where `id` is the internally generated Moab job id number and `.poweron` and `.poweroff` are suffixes appended to the job id number that represent Moab's on and off commands sent to Power RMs.

## Green Policies

Moab provides green policies that automate power management for idle compute nodes, which an administrator can modify and/or configure to control the power state of compute nodes not always in use. These policies allow Moab to dynamically control the power state of compute nodes between the active running state or power-on nodes that may be needed. It also allows Moab to power-off nodes that are idle and wasting energy. Which power state such compute nodes enter depends entirely on the commands the administrator configures and/or modifies in a power RM's scripts and, for the newer

Moab+MWS method, on configuration information specified for each MWS RM power management plug-in instance.

The green policies maintain a green idle node pool, the size of which the administrator configures. As jobs start and use idle nodes from the pool, Moab replenishes the pool by performing an `on` command on those compute nodes on which it previously had performed an `off` command, thus bringing them into the idle node pool as they enter into an active running state. When jobs finish and the pool has excess idle nodes, Moab performs an `off` command on the excess nodes, which removes them from the idle pool. Thus, Moab maintains a pool of available idle nodes for immediate use by submitted jobs and reduces energy consumption by powering off any idle nodes in excess of the pool size.

## Theory of Operation

Moab itself operates the same regardless of the method of power management, Moab-only or Moab+MWS, chosen. This is especially true for the green policies as Moab simply uses the configured power management method to carry out the policies. In order to know how to configure the different parts and components of each power management method so they work well together, it is necessary for a site administrator to understand how the power management methods work; that is, how the components work together to implement a power management method.

### Moab-only Method

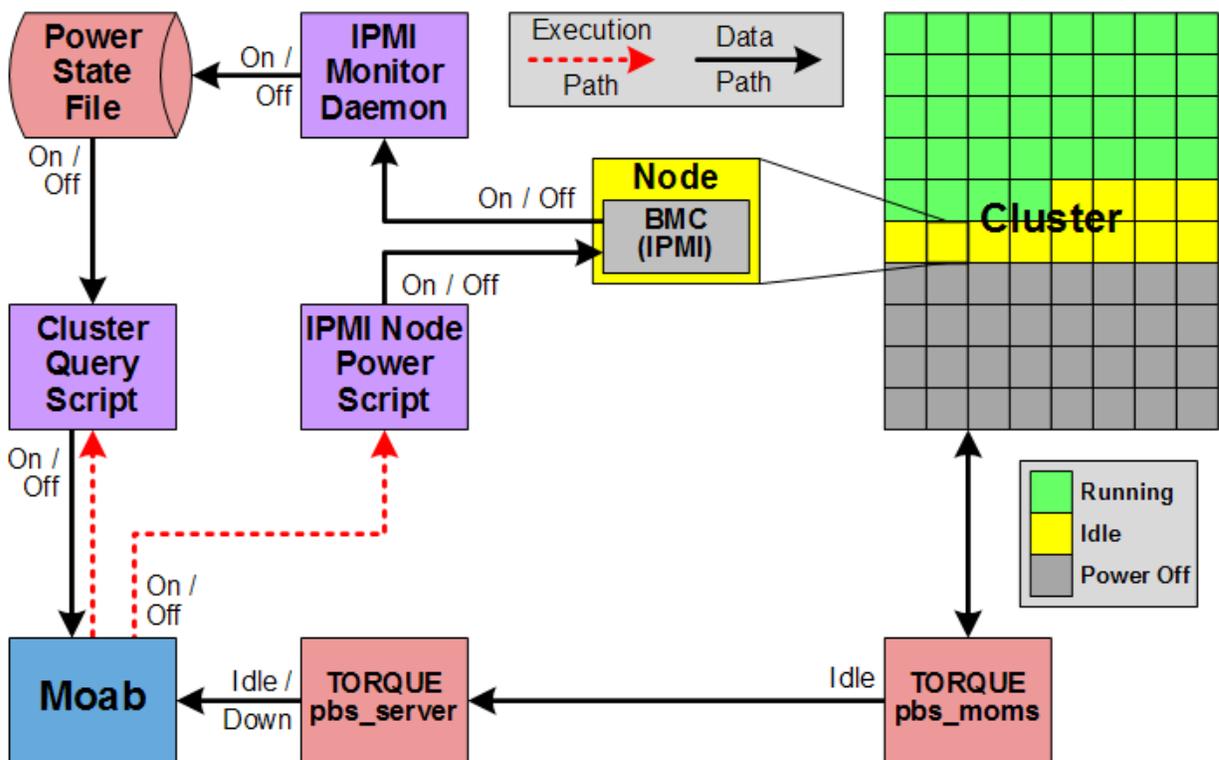
The Moab-only method has a Power RM composed entirely of Python-based scripts. The script must maintain a Power Query daemon that queries the power state of all compute nodes and saves their state for Moab to query, the actual power state query Moab runs to find out the current power state of all compute nodes, and a power state control that places compute nodes into the state of `on` so Moab can schedule jobs to them or into the state of `off` so energy consumption is minimized and operational costs reduced. The administrator determines what the actual power state Moab's `off` represents by configuring the `off` command in the power management control script with the actual hardware vendor-supplied command that effects the desired power state (remember, Moab is not aware of actual power states).

The list below enumerates the advantages and disadvantages of the Moab-only method.

- Advantages
  - Do not have to run the MWS service and its MongoDB database.
  - Power management command scripts execute as Moab system jobs.
  - Ability to customize the node power and cluster query power management scripts
    - For more information on how to specify the node power control script, see the [NODEPOWERURL](#) parameter.
    - For more information on how to specify the power cluster query script, see the [CLUSTERQUERYURL](#) parameter.
  - Moab power control using `mnodectl -m power=[on|off] <nodelist>`.
    - For more information on how to diagnose power states, see [mdiag -n](#).

- Disadvantages
  - More complex scripts to customize.
  - Only global power management control (no partition-based or node-based).
  - Heterogeneous compute node hardware from different vendors requires more modification of the control and query scripts.
  - Reference scripts not scalable (did not take advantage of Python multi-threading).
  - Administrator must maintain complex scripts that must maintain the entire cluster query information.

The following architecture diagram shows the Moab-only architecture and what occurs between its components.



The Python-based IPMI Monitor daemon script running in the background periodically polls the power state of all compute nodes through IPMI using the command customized by the administrator. As it gathers power state information, it saves the information in a text file in a specific format understood by Moab (binary power state). In order to prevent race conditions, it actually writes to a temporary file and then moves the temporary file on top of the permanent file (not shown).

When Moab starts a scheduling cycle/iteration, it directly executes the power RM's Python-based Cluster Query script that reads the permanent text file and delivers the compute node power states to Moab. Moab then performs the scheduling cycle and based on green policies and the state of the HPC cluster will run the IPMI Node Power script as a Moab system job to perform an `on` or `off` (which may be

something different than a `power off`) command using the actual commands customized by the administrator in the script.

## Moab+MWS Method

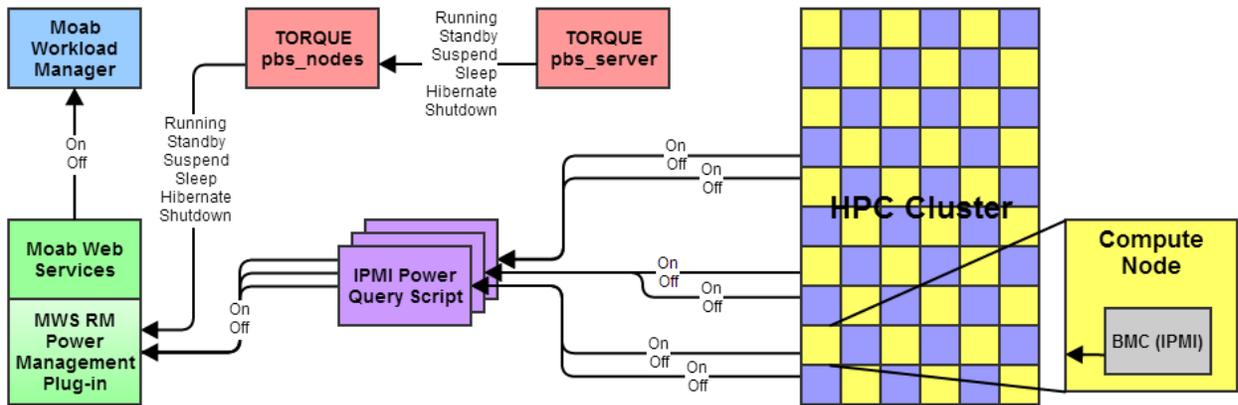
The Moab+MWS method has a Power RM composed of a MWS RM plug-in that encapsulates all power management logic, which itself uses the TORQUE `pbsnodes` command to effect compute node power state changes into low-power and no-power states of standby and suspend, and hibernate and shutdown, respectively, as well as the IPMI Node Power script to effect compute node power on, power off (pull the plug) and awaken (resume active running state from low-power state). The Power RM Power Management plug-in also performs the power query daemon function identified in the Moab-only method using its built-in power management logic, thus handling more actual power states and allowing much better power control than the Moab-only method offers.

The advantages and disadvantages of the Moab+MWS-based method are enumerated below.

- Advantages
  - More power states to choose from.
    - Low-power states of standby and suspend.
    - No-power states of hibernate and shutdown.
    - On and Off (pull the plug) power states still available.
  - TORQUE power control of low-power and no-power states using `pbsnodes -m <state> <nodelist>`.
    - You can view node power states with the `pbsnodes` command.
  - Power management command scripts execute as Moab system jobs.
  - Much simpler `moab.cfg` [customization and maintenance](#).
  - Global, partition-based, and node-based granularity for power management control.
  - Heterogeneous compute node hardware from different vendors handled by creating multiple instances of MWS RM power management plug-in with different configurations.
  - Reference scripts are scalable (use Python multi-threading).
  - The MWS RM architecture is easier to support DRAC, ILO, and other protocols.
- Disadvantages
  - Must run the MWS service and its MongoDB database.
  - Configuration of the MWS RM Power Management plug-in and possible multiple instances.

The following architecture diagrams show the Moab+MWS-based method architecture and what occurs between its components.

The diagram below illustrates power state query:

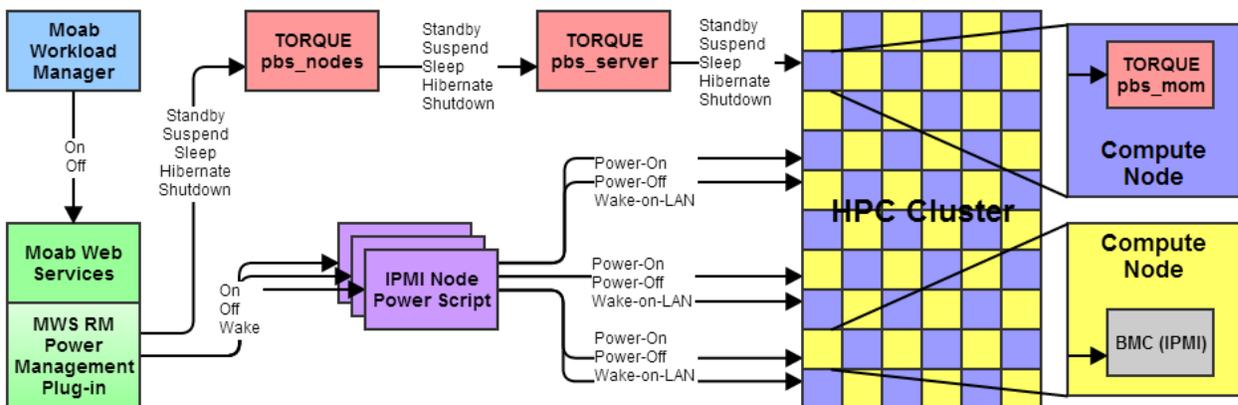


The MWS RM power management plug-in runs the multi-threaded Power Query script for sets of compute nodes which obtain their actual power state through IPMI, or more specifically, a hardware vendor's IPMI implementation (e.g., Dell DRAC, HP iLO, etc), which the RM plug-in saves. It also runs the TORQUE `pbsnodes` command to obtain the low-power or no-power states that may have been set via TORQUE earlier (`pbs_server` retains knowledge of any previous command to set a node's power state to one of the low-power or no-power states).

Note it is quite possible for IPMI to report `off` and TORQUE to report `hibernate` or `shutdown`, both of which indicate a compute node has no power, and for IPMI to report `on` and TORQUE to report `standby` or `suspend`, both of which indicate a compute node is in a low-power state from which it can be quickly awakened. It is also possible for IPMI to report `on` and TORQUE to report `hibernate` or `shutdown`, which can indicate a booting node that has not yet started the TORQUE `pbs_mom` daemon or a node hibernating or shutting down that has not yet powered off. The MWS plug-in's power management logic reconciles the IPMI and TORQUE reports to produce a single `on` or `off` understood by Moab, which it passes to MWS.

When Moab queries MWS for the current state information of compute nodes at the start of a scheduling cycle/iteration, MWS passes all node information including the binary power on/off Moab understands and the TORQUE node state, at which point Moab has the information it needs to perform green policy-based automated power management.

The diagram below illustrates Moab+MWS power state control interactions.



When Moab detects a condition that requires changing the power state of a compute node, usually as a result of green policies, it performs the appropriate `on` or `off` command as a system job that sends the command to MWS with a list of the host names of compute nodes that should enter an appropriate power state.

MWS interacts with the appropriate MWS RM power management plug-in for each compute node and passes it the `on` or `off` command. For the `off` command, the plug-in examines its configuration of what `off` means and passes the configured standby, suspend, hibernate, or shutdown command to the TORQUE `pbsnodes` command, or passes the configured `off` command to the Node Power script.

If the RM plug-in executes the TORQUE `pbsnodes` command for the configured power state and requested list of compute node host names, it sends the command to the `pbs_server`, which passes the command to each compute node's `pbs_mom` daemon. The `pbs_mom` executes software to place the node into the requested state. The `pbs_server` daemon keeps the requested state in a file for each compute node, which it passes on to the MWS RM power management plug-in as part of a node update report.

**i** In clusters where there is a TORQUE `pbs_server` and `pbs_mom` on the same machine, the administrator should set the **POWERPOLICY** to **STATIC** on this node, because the `pbs_server` should not be powered down. If the `pbs_server` is powered down, Moab will be unable to get cluster query updates from all `pbs_moms` managed by that that `pbs_server`.

On all TORQUE nodes where `pbs_moms` are running, the `pbs_mom` must be configured to auto-start after being rebooted. If the `pbs_mom` isn't auto-started, the `pbs_server` will not be able to determine when it has been powered up and entered an idle state, and therefore won't have the ability to inform Moab on a cluster query the node is idle. Refer to [Startup/Shutdown service script for TORQUE/Moab \(OPTIONAL\)](#) on page 2836 for Torque/Moab for details on how to have the `pbs_mom` auto-start on boot.

When the RM plug-in executes the Node Power script for the configured `off` power state and requested list of compute node host names, the script executes its IPMI `on` command (whatever the administrator configured in the script) that tells the node's BMC to power off the node.

When the RM plug-in receives the `on` command from Moab via MWS, it checks the internal power state of each compute node in the requested list of compute node host names. If the internal power state is standby or suspend, the script executes its IPMI `wake` command (whatever the administrator configured in the script) that tells the node's BMC to bump the node into the active running state; otherwise, the script executes its IPMI `off` command (whatever the administrator configured in the script) that tells the node's BMC to power on the node.

**i** Some operating systems require the Wake-on-LAN bit to be enabled using a tool like `ethtool`. Also, Wake-on-LAN packets may be blocked by the router, but not always.

In this manner, the MWS RM power management plug-in queries the actual power state of individual compute nodes and returns to Moab the simple binary on/off state it understands for scheduling jobs to compute nodes. Likewise, Moab controls the actual power state of individual compute nodes using only its simple binary on/off command. This method of simple command and simple job-scheduling-ability state enables Moab to remain scalable and responsive for automatic power management control using green policies.

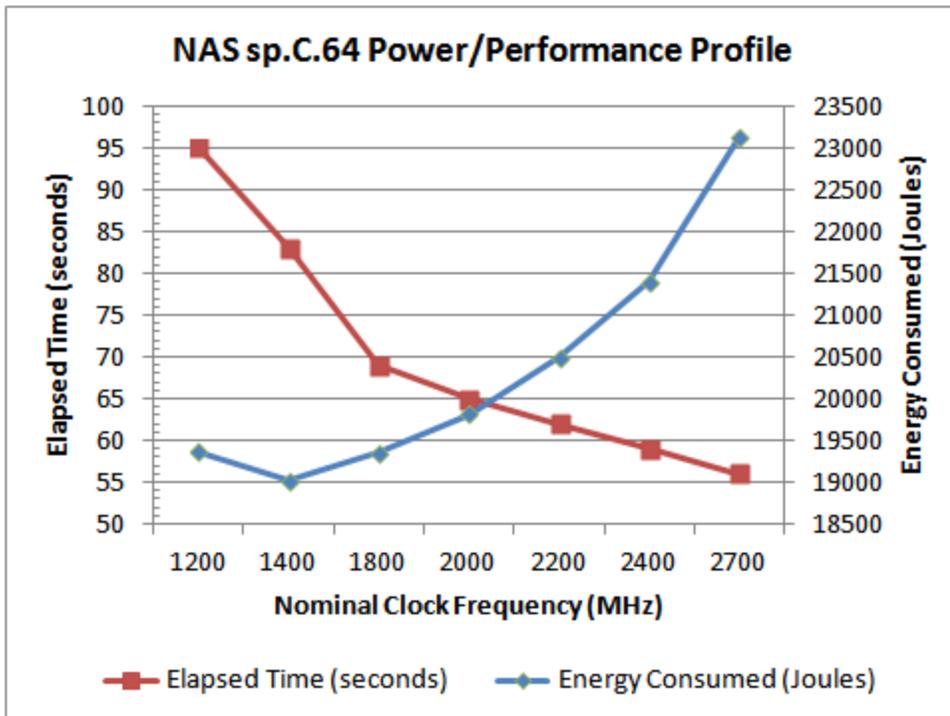
## Active Node Power Management

Moab 8.0 and TORQUE 5.0 introduce support for active node power management; that is, the management of energy consumption while a compute node is running a job, which the new CPU Clock Frequency Control feature provides.

The amount of energy consumption savings achievable through the CPU Clock Frequency Control feature is application-dependent. For example, memory, I/O, and/or network-bound applications, especially memory-bound applications, can often drop the clock frequency of their compute nodes' processors and still have the same execution time even though the compute nodes consume less power. Several studies have shown common power savings of 18-20% and one study showed one application saving 30% on power consumption, all of which translate directly into operational cost savings.

### Power/Performance Profiling

To determine whether a lower clock frequency will produce energy consumption savings, applications must be profiled; that is, a job running a particular application with the same or equivalent data must be run at different clock frequencies while measuring the energy consumption of the job's compute node. Each pair of frequency/energy consumption data points are plotted in a chart to show the application's power performance profile. The charts below are an example of two such profiles for two NAS benchmark HPC applications.



The intersection of the two lines has no meaning as each line has its own vertical scale, either on the left or the right as noted!

Note both applications do not consume the least energy (vertical dashed green line) when running at the lowest clock frequency, which demonstrate the importance of profiling applications to determine the nominal clock frequency at which energy consumption is the lowest. The charts amply illustrate why a simplistic policy of using the lowest clock frequency is not the best policy when a site's objective is the least energy consumption possible.

If the least energy consumption is not a site's primary objective, but running jobs in a manner that balances energy consumption and job execution time, a power/performance profile chart is very useful to determine the clock frequency that meets a balanced objective. For example, the vertical dashed purple line on the right chart shows that running the `bt.C.64` application at 1800 MHz has an increase in energy consumption of ~1% over the minimal energy consumption possible (vertical dashed green line) but results in a ~10% drop in execution time; a possibly very good trade-off!

Obviously, if a site's primary objective is to complete a job as fast as possible but do so saving energy where possible, profiling memory-bound and other bound applications can clearly show the lowest clock frequency at which the application takes longer to execute. The site would then institute a policy that the application should run at the next highest frequency to fulfill the twin objectives of job performance and energy consumption minimization.

For more information about the CPU clock frequency job submission option, see [CPUCLOCK resource manager extension](#) of [msub -l](#).

### *Job Templates*

Most users will not care or want to know about clock frequency control, so administrators can use a job template to specify the CPU clock frequency at which a particular recurring job should execute. A clock frequency specified on a job template overrides a clock frequency given on the job submission command line or inside a job script file with TORQUE PBS commands. This order of precedence allows an administrator to control clock frequency for commonly used applications and jobs based on site policies and objectives.

For more information about using a CPU clock frequency job submission option in job templates, see the [CPUCLOCK job template extension attribute](#).

## Idle node power management

Moab has so-called green policies that together configure Moab to manage and maintain a pool of idle nodes in an active running state so it can immediately schedule jobs to them. When Moab does so and diminishes the pool's idle compute node quantity, it powers on compute nodes by performing an `on` command for nodes in a powered-down state (actually, in a low-power or no-power state) to bring them on-line in order to replenish the pool of idle nodes up to its configured size. When jobs end and the idle node exceed the configured idle node pool size and there are no jobs to run on the now-idle nodes, Moab will power off excess idle nodes by performing an `off` command. In this manner, Moab achieves a site's power management and energy consumption objectives through the configured green policies.

See the Moab-only Method Architecture diagram above to see the color-coded compute nodes in the diagram's cluster illustrating Moab's green idle node pool management. The green nodes represent nodes running jobs, the yellow nodes are idle nodes in a green pool of size 12, and the gray nodes represent `off` nodes. Note Moab does not know what actual power state `off` means; what it means will

be based on command customization inside Moab-only method scripts or Moab+MWS plug-in configuration information.

In order to perform green policy management of an idle node pool, Moab must first be configured to use either the Moab-only or the Moab+MWS method of power management. It is *best practice* to configure power management first and test its configuration before configuring green policies. Thus, if power management is misconfigured, an administrator will know it is the power management configuration and/or scripts and not the green computing policies that are incorrect. If the manual power management commands for the configured power management method work, green computing will work using the configured power management method. For information on how to configure each power management method in Moab, see [Enabling green computing on page 736](#).

## Green Policy Configuration

There are several green policies that affect how Moab performs green idle node pool management using automated power management operations. The policies are configured in the same manner regardless of the power management method used, whether Moab-only or Moab+MWS. The other sections of this chapter describe how to configure green policies that manage the idle node pool for site energy management objectives.

### Related topics

- [Enabling green computing on page 736](#)
- [Deploying Adaptive Computing IPMI scripts on page 733](#)
- [pbsnodes on page 2705](#)

## How-to's

### Deploying Adaptive Computing IPMI scripts

#### Context

If you want to enable green computing on your system using the Adaptive Computing supplied IPMI reference scripts, follow the steps here. The IPMI scripts provided are meant as a reference for you to configure the solution to your environment, but can also be used as-is.

#### Prerequisites

- OpenIPMI and ipmitool must be installed and working.
- All nodes must have the same IPMI username and password.
- You must know the IPMI host names and/or IPMI IP addresses of your nodes.
- Python must be installed. The provided IPMI scripts were developed using Python 2.6.5.
- You must identify your Moab home directory. These instructions assume the default Moab home directory of `/opt/moab`.
- You must identify your Moab tools directory. These instructions assume the default Moab tools directory of `/opt/moab/tools`.

## To deploy the Adaptive Computing IPMI scripts

1. Edit the `/opt/moab/tools/ipmi/config.py` script:
  - a. Set **self.ipmiuser** to the IPMI username for your nodes.
  - b. Set **self.ipmipass** to the location of the IPMI password file (`/opt/moab/passfile.txt` by default).

**i** The permissions for the directory and the password file itself should be set so that they can be read only by root or the Moab user running the script.

- c. Set **self.homeDir** to your Moab home directory.
  - d. If desired, change the **self.pollInterval** value. This is the interval, in seconds, between polls from the IPMI monitoring script.
  - e. The **self.ipmifile** value is the name of a temporary file where the cluster query information is stored. You can change this or leave it alone.
  - f. The **self.bmcaddrmap** value is the filename for the Moab node name/IPMI mapping. The file must exist in the Moab home directory and will be created in the next step.
2. Create a `node-bmc.txt` file in the Moab home directory. The file must contain a space-delimited list of Moab node names that map to the IPMI host names or IP address. For Example:

```
node01 node01_ipmi # For all three of these entries, the first value is the
node02 node02_ipmi # node name as Moab knows it. The second value is either
node03 10.1.1.1    # the node IPMI name or IPMI IP address.
```

3. Configure the `moab.cfg` file for green computing as described in [Enabling green computing](#). Use the `ipmi.mon.py` script for the `CLUSTERQUERYURL` and the `ipmi.power.py` script for the `NODEPOWERURL`.
4. Restart Moab and verify green computing is working correctly. If you encounter trouble, see the [Troubleshooting green computing](#) topic for help.

### Related topics

- [Enabling green computing on page 736](#)
- [Troubleshooting green computing on page 742](#)
- [Adjusting green pool size on page 740](#)
- [Handling power-related events on page 740](#)
- [Maximizing scheduling efficiency on page 741](#)

## Choosing which nodes Moab powers on or off

### Context

Moab can use the GREENPOOLPRIORITYF function to determine which nodes to power on or off. The [PRIORITY node allocation policy](#) is used to determine which nodes to allocate workload to. When Moab can no longer allocate workload to available nodes, it begins to power nodes on in the order specified by the GREENPOOLPRIORITYF function.

### To choose which nodes Moab powers on or off

1. Set a GREENPOOLPRIORITYF function to describe which order nodes should be selected for power on/off actions. GREENPOOLPRIORITYF uses the [PRIORITY node allocation policy](#) options and syntax.

```
GREENPOOLPRIORITYF '10*RANDOM'
```

*This tells Moab to randomly choose a node to power on to meet workload demands, and to randomly choose an idle node to power off to meet the [MAXGREENSTANDBYPOOLSIZE](#) goal.*

### To choose which nodes Moab allocates jobs to

1. Set a PRIORITY node allocation policy that uses power as the major factor. This causes Moab to allocate jobs to nodes that are already powered on. When no nodes are available to meet this policy, Moab uses the GREENPOOLPRIORITYF function to turn on nodes that are powered off.

```
NODEALLOCATIONPOLICY PRIORITY
NODECFG[DEFAULT] PRIORITYF='10000*POWER + 10*RANDOM'
```

*The nodes with the highest priority for workload are the nodes that are powered on. After that, Moab randomly allocates workload.*

### Related topics

- [Adjusting green pool size on page 740](#)
- [Maximizing scheduling efficiency on page 741](#)

## Enabling green computing

### Context

There are two ways to do green computing in Moab. With just Moab, nodes can be turned on or off. With [MWS](#), however, you can put nodes into several low-power states. The MWS solution is also more scalable. The supported low-power states are:

- Running
- Standby
- Suspend
- Hibernate
- Shutdown

Nodes cannot be moved from one low-power state to another. The node must go from low-power to running, and then to the new low-power state.

### To enable green computing with Moab and MWS

1. Edit `moab.cfg` to use MWS for green computing:
  - a. Configure the [POWERPOLICY](#) attribute of the [NODECFG](#) parameter. The default value is *STATIC*. Set it to *OnDemand*.
  - b. Set the resource manager type as *MWS*
  - c. Set `FLAGS=UserSpaceIsSeparate` for the MWS resource manager.
  - d. Point `BASEURL` to your MWS server.

```
NODECFG [DEFAULT]      POWERPOLICY=OnDemand
RMCFG [mws]            TYPE=MWS
RMCFG [mws]            FLAGS=UserSpaceIsSeparate
RMCFG [mws]            BASEURL=http://localhost:8080/mws
```

2. Configure the MWS [Power Management Plugin on page 1758](#).

### To enable green computing with just Moab

1. Edit `moab.cfg` to enable green computing. There are four things you must configure for basic functionality of green computing:
  - a. Configure the `POWERPOLICY` attribute of the `NODECFG` parameter. The default value is `STATIC`. Set it to `OnDemand`.
  - b. Configure a power provisioning resource manager to be `TYPE=NATIVE` and `RESOURCE_TYPE=PROV`. The resource type of `PROV` means the RM works only with node hardware and not workloads.
  - c. Configure a `CLUSTERQUERYURL` attribute of the power provisioning RM to point to the power query script you'd like to use. Moab uses this script to query the current power state of the nodes. `CLUSTERQUERYURL` is traditionally used as a workload query but is also used by green computing for the node power state query. Adaptive Computing provides a reference [IPMI script](#) you can use.
  - d. Configure a `NODEPOWERURL` attribute of the power provisioning RM to point to the power action script you'd like to use. Moab uses this script to turn nodes on or off. Adaptive Computing provides a reference [IPMI script](#) you can use.

```
NODECFG[DEFAULT] POWERPOLICY=OnDemand
RMCFG[ipmi] TYPE=NATIVE RESOURCE_TYPE=PROV
RMCFG[ipmi] CLUSTERQUERYURL=exec://$TOOLS_DIR/ipmi/ipmi.mon.py
RMCFG[ipmi] NODEPOWERURL=exec://$TOOLS_DIR/ipmi/ipmi.power.py
```

### Sample moab.cfg for green computing

Below is a sample `moab.cfg` configuration file of a green computing setup using the Adaptive Computing IPMI scripts.

```

#####
#
# Use 'mdiag -C' to validate config file parameters
#
#####

SCHEDCFG[Moab]      SERVER=myhostname:5150
ADMINCFG[1]         USERS=myusername,root
TOOLS DIR           /$HOME/tools
LOGLEVEL            1

#####
#
# Basic Resource Manager configuration
#
# For more information on configuring a Resource Manager, see:
# docs.adaptivecomputing.com
#
#####

RMCFG[local]       TYPE=NATIVE
RMCFG[local]       CLUSTERQUERYURL=exec://$HOME/scripts/query.resource
RMCFG[local]       WORKLOADQUERYURL=exec://$HOME/scripts/query.workload

RMCFG[local]       JOBSUBMITURL=exec://$HOME/scripts/submit.pl
RMCFG[local]       JOBSTARTURL=exec://$HOME/scripts/job.start
RMCFG[local]       JOBCANCELURL=exec://$HOME/scripts/job.cancel
RMCFG[local]       JOBMODIFYURL=exec://$HOME/scripts/job.modify
RMCFG[local]       JOBREQUEUEURL=exec://$HOME/scripts/job.requeue
RMCFG[local]       JOBSUSPENDURL=exec://$HOME/scripts/job.suspend
RMCFG[local]       JOBRESUMEURL=exec://$HOME/scripts/job.resume

#####
# GREEN configuration:
#####
# Turn on "green" policy. (This is the policy that enables green computing).
# Here we are doing it for all nodes, but it can be controlled on a node-by-node
basis
# Default is STATIC, which means green computing is disabled.
#NODECFG[DEFAULT]  POWERPOLICY=STATIC
NODECFG[DEFAULT]  POWERPOLICY=OnDemand

# Use the MWS RM and the MWS power management plugin for power provisioning
# and power state.

```

```

RMCFG [mws] TYPE=MWS
RMCFG [mws] FLAGS=UserSpaceIsSeparate
RMCFG [mws] BASEURL=http://localhost:8080/mws

# We want green policy to work so it allocates jobs to compute nodes already
# powered on and will power on powered-off compute nodes only when there are
# no powered-on compute nodes available. This requires using the PRIORITY
# node allocation policy with a PRIORITYF function that has the POWER variable
# as the greatest contributing factor to the function (1 = powered-on,
# 0 = powered-off).
# If we want all compute nodes to operate under green policy, we can assign
# the PRIORITYF function to the default node configuration, which is easier
# than assigning it to individual compute nodes. If only some compute nodes
# should operate under green policy, then the PRIORITYF function must be
# configured for the individual nodes. Note the POWER variable must be the
# largest factor in the function below; it is assigned the largest multiplier,
# which should be greater than the sum of all other factors! Doing so forces
# Moab to use all eligible powered-on nodes for workload placement before
# powering on any eligible powered-off nodes.

# Enable PRIORITYF functionality
#NODEALLOCATIONPOLICY PRIORITY

# Use a priority function that uses power as the major factor (plus some other
# imaginary factors)
#NODECFG[DEFAULT] PRIORITYF='1000000*POWER + 1000*factor2 + 100*factor3...'
# Use a priority function where power is the only factor.
#NODECFG[DEFAULT] PRIORITYF='10000*POWER'
# Use a priority function that adds some randomness but uses power as the major
# factor.
#NODECFG[DEFAULT] PRIORITYF='10000*POWER + 10*RANDOM'

# Set a priority function that specifies the order nodes should be chosen to power
# up/down. By default, Moab will start at the top of the node list and go down.
# Some
# installations want to rotate power cycles among nodes in a different order.
# The configuration below forces Moab to power on/off random nodes, which
# eventually guarantees all nodes occasionally go through a power cycle.
#GREENPOOLPRIORITYF '10*RANDOM'

# Ensure we are recording power management events
# (powering on and off nodes are recorded as "node modification" events).
#RECORDEVENTLIST +NODEMODIFY

# Set the size of the standby pool. This is the number of idle nodes that will
# be powered on and idle. As the workload changes, Moab turns nodes on
# or off to try to meet this goal.
# Default value is 0
MAXGREENSTANDBYPOOLSIZE 5

# Set the length of time that it takes to power a node on/off. This will be the
# walltime of the system job that performs the power operation and should be the
# maximum expected time. If Moab detects (via the power RM) that the power
# operations have all completed, the system job will finish early.
# Default value is 10 minutes (600)
PARCFG[ALL] NODEPOWEROFFDURATION=600
PARCFG[ALL] NODEPOWERONDURATION=600
# Set the length of time a node should remain idle before it is powered off.
# This prevents Moab from immediately powering off nodes that have just finished
# a job. Increasing this number should decrease power on/off thrashing
# This should be set higher than NODEPOWEROFFDURATION and/or NODEPOWERONDURATION

```

```
NODEIDLEPOWERTHRESHOLD 660
```

```
# If a node fails to power on, we need to remove it from the available nodes so
# Moab won't keep [re-]trying to power it on. Do this by setting a reservation
# on the failed node to give time for manual investigation.
#RMCFG[torque] NODEFAILURERSVPROFILE=failure
#RSVPROFILE[failure] DURATION=3600
```

## Related topics

- [Deploying Adaptive Computing IPMI scripts on page 733](#)
- [Choosing which nodes Moab powers on or off on page 735](#)
- [Adjusting green pool size on page 740](#)
- [Handling power-related events on page 740](#)
- [Maximizing scheduling efficiency on page 741](#)
- [Troubleshooting green computing on page 742](#)
- [Power Management Plugin on page 1758](#)

## Adjusting green pool size

### Context

The MAXGREENSTANDBYPOOLSIZE parameter allows you to allocate the number of nodes to keep powered on in the standby pool. This is the number of idle nodes that are allowed be powered on and idle. As the workload changes, Moab turns nodes on or off to try to meet this goal. The default value is 0.

### To adjust the green pool size

1. Modify the MAXGREENSTANDBYPOOLSIZE parameter with the number of nodes you want Moab to keep powered on for the standby pool.

```
MAXGREENSTANDBYPOOLSIZE 10
```

*Moab keeps up to 10 idle nodes powered on to be kept on standby.*

### Related topics

- [Maximizing scheduling efficiency on page 741](#)
- [Choosing which nodes Moab powers on or off on page 735](#)

## Handling power-related events

### Context

Power actions are considered [NODEMODIFYURL](#) events and are not recorded by default, but you can configure Moab to include power-related events in the logs. Also, if a node fails to turn on (or off), it's best to associate a reservation on the failed node so that Moab won't keep trying to perform the power action over and over.

## To configure Moab to record power-related events

1. Modify the **RECORDEVENTLIST** parameter.

```
RECORDEVENTLIST +NODEMODIFY
```

*Power-related events are logged to the Moab log file.*

## To put a reservation on a node that fails to perform a power action

1. Configure the **NODEFAILURERSVPROFILE** attribute of **RMCFG** and create an **RSVPROFILE** with a high duration.

```
RMCFG[torque] NODEFAILURERSVPROFILE=failure
RSVPROFILE[failure] DURATION=3600
```

*Nodes that fail to power on or off have a 1-hour reservation placed on them.*

### Related topics

- [RECORDEVENTLIST](#) on page 1026
- [Event Logs](#) on page 679

## Maximizing scheduling efficiency

### Context

When considering whether to power a node on or off, Moab can take into account the amount of time that it takes to power on or power off the node. With this information, Moab can keep an idle node powered on if it knows that workload in the queue will be ready for the node in less time that it takes to power off/power on the node.

Moab can also wait to shut down nodes after they've been idle for a specific amount of time.

### To specify node power on/power off duration

1. Modify the **NODEPOWERONDURATION** and **NODEPOWEROFFDURATION** attributes of **PARCFG** with the maximum amount of time it takes for your nodes to power on/power off. Make sure to use the keyword **ALL** for the resource manager name to avoid cases where Moab won't consider the power on/off duration for a node before making a power action decision.

```
PARCFG[ALL] NODEPOWERONDURATION=2:00
PARCFG[ALL] NODEPOWEROFFDURATION=2:00
```

*If a node goes idle and has to wait for workload, Moab will not power off the node if the workload will be available within 4 minutes or less.*

### To shut down on nodes after they've been idle for a specified time

1. Modify the **NODEIDLEPOWERTHRESHOLD** parameter with the duration (in seconds) you want Moab to wait before shutting down an idle node. The default value is 60 seconds. Increasing the number should decrease power on/off thrashing. This should be set higher than **NODEPOWERONDURATION**

and/or **NODEPOWEROFFDURATION**.

```
NODEIDLEPOWERTHRESHOLD 300
```

*Moab will wait 5 minutes before shutting down a node that has become idle.*

#### Related topics

- [Adjusting green pool size on page 740](#)
- [Choosing which nodes Moab powers on or off on page 735](#)

## Putting idle nodes in power-saving states

### Context

When nodes exceed their idle threshold limits, the default behavior is to turn the nodes off. With the **NODEIDLEPOWERACTION** parameter, you can choose which power-saving state to put idle nodes into. This parameter is configured at the partition level. Configuring it for the **ALL** partition effectively makes it a global parameter.

### To specify what to do with idle nodes

1. Modify the **NODEIDLEPOWERACTION** parameter.

```
NODEIDLEPOWERTHRESHOLD 300
PARCFG[ALL] NODEIDLEPOWERACTION SLEEP
```

*All nodes that are idle for more than 5 minutes are put into a sleep state.*

#### Related topics

- [Enter xrefs to related topics here. Use a simple ul style and apply the li.links style to the list items.]
- 

## Troubleshooting green computing

### Context

If you've enabled green computing and are having trouble, here are some tips that can help you determine the cause of the issues you encounter. These tips are specifically for the [Adaptive Computing supplied IPMI scripts](#), but can be generalized for whatever power management solution you use. Simply substitute your power management system, power query script (as specified by **CLUSTERQUERYURL**), and power action script (as specified by **NODEPOWERURL**) where appropriate.

### Verify your IPMI access

1. Use the `ipmitool` command to verify you have access to the IPMI interface of your nodes. Try getting the current power state of a node. The syntax is `ipmitool -I lan -H <host> -U <IPMI username> -P <IPMI password> chassis power status`.

```
$ ipmitool -I lan -H qt06 -U ADMIN -P ADMIN chassis power status
Chassis Power is off
```

### Verify the power query (CLUSTERQUERYURL) script is working

1. Execute the `ipmi.mon.py` script (should be found in `/<MOABHOMEDIR>/tools/ipmi`) to start the monitor.

```
$ cd /opt/moab/tools/ipmi
$ ./ipmi.mon.py
```

2. Execute the script again. The following is an example of the expected output:

```
$ ./ipmi.mon.py
qt09  GMETRIC[System_Temp]=27 GMETRIC[CPU_Temp]=25 POWER=on State=Unknown
qt08  GMETRIC[System_Temp]=31 GMETRIC[CPU_Temp]=25 POWER=on State=Unknown
qt07  GMETRIC[System_Temp]=30 GMETRIC[CPU_Temp]=29 POWER=on State=Unknown
qt06  GMETRIC[System_Temp]=Disabled GMETRIC[CPU_Temp]=Disabled POWER=off
State=Unknown
```

*If the **POWER** attribute is not present the script is not working correctly.*

### Verify the power action (NODEPOWERURL) script is working

1. Execute the `ipmi.power.py` script (should be found in `/<MOABHOMEDIR>/tools/ipmi`) to see if you can force a node to power on or off. The syntax is `ipmi.power.py <node>,<node>,<node>... [off|on]`

```
$ /opt/moab/tools/ipmi/ipmi.power.py qt06 off
```

*This example is trying to power off a node named qt06.*

2. Verify the machine's power state was changed to what you attempted in the previous step. You can do this remotely via two methods:
  - a. If the [cluster query script](#) is working, you can use that to verify the current power state of the node.
  - b. If you have [IPMI access](#), you can use the `ipmitool` command to verify the current power state of the node.

## Verify the scripts are configured correctly

1. Run the `mdiag -R` command to verify your IPMI resource manager configuration.

```
$ mdiag -R -v
RM[ipmi]      State: Active  Type: NATIVE  ResourceType: PROV
Timeout:      30000.00 ms
Cluster Query URL: exec://$TOOLSDIR/ipmi/ipmi.mon.py
Node Power URL:  exec://$TOOLSDIR/ipmi/ipmi.power.py
Objects Reported: Nodes=3 (0 procs)  Jobs=0
Nodes Reported: 3 (N/A)
Partition:     SHARED
Event Management: (event interface disabled)
RM Performance: AvgTime=0.05s MaxTime=0.06s (176 samples)
RM Languages:  NATIVE
RM Sub-Languages: NATIVE
```

2. Run the `mdiag -G` command to verify that power information is being reported correctly.

```
$ mdiag -G

NodeID      State      Power      Watts      PWatts
qt09        Idle       On          0.00       0.00
qt08        Idle       On          0.00       0.00
qt07        Idle       Off         0.00       0.00
```

## Verify the scripts are running

1. Once green is configured and Moab is running, Moab should start the power query script automatically. Use the `ps` command to verify the script is running.

```
$ ps -ef | grep <CLUSTERQUERYURL script name>
```

*If this command does not show the power query script running then your settings in `moab.cfg` aren't working.*

## Verify Moab can power nodes on or off

1. Use the `mnodectl` command to turn a node on or off. The syntax is `mnodectl -m power=[off|on] <node>`.

```
mnodectl -m power=off qt06
```

*Moab should turn off the node named qt06.*

- a. Moab generates a system job called `poweron-<num>` or `poweroff-<num>` job as shown in [showq](#). The system job calls the `ipmi.power.py` (**NODEPOWERURL**) script to execute the command.
- b. Moab waits until the cluster query reports the correct data. In this case, the `ipmi.power.py` script reports that the power attribute has changed.
- c. Moab does not change the power status based on the power script return code. Rather, Moab completes the system power job when it detects the power attribute has changed as indicated by the cluster query script.

Related topics

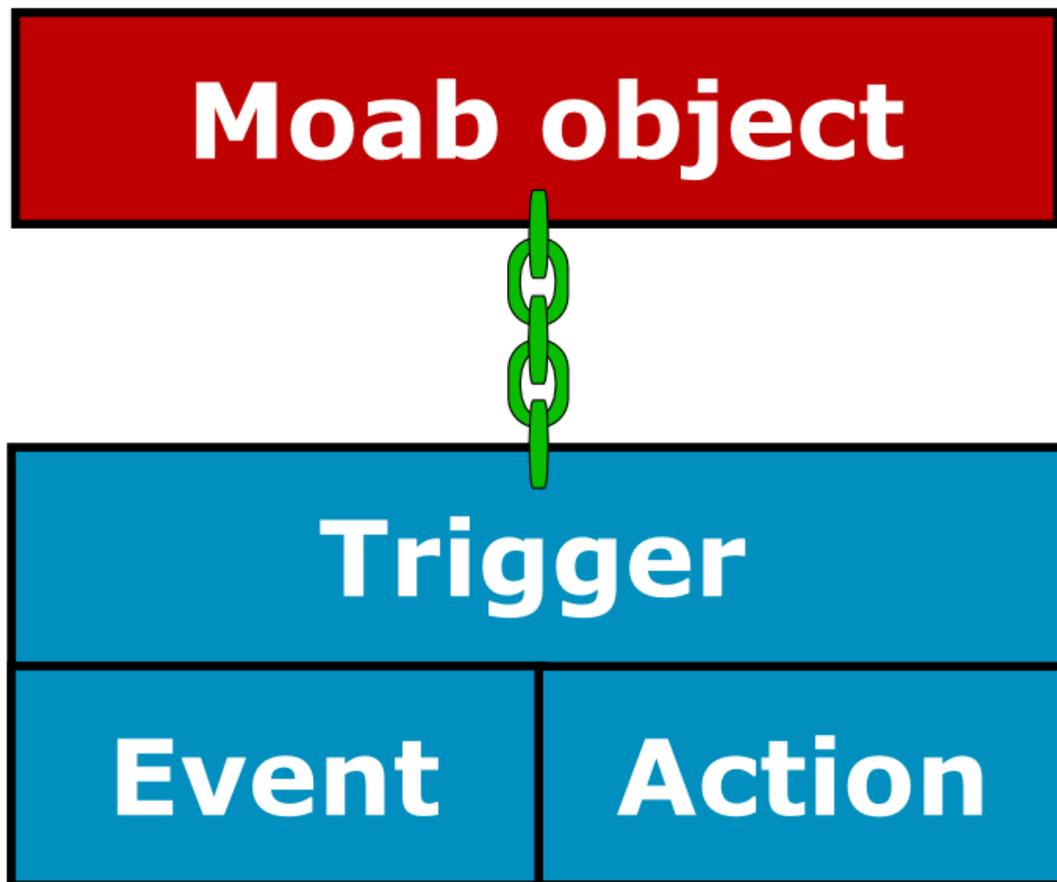
- [Enabling green computing on page 736](#)
- [Deploying Adaptive Computing IPMI scripts on page 733](#)

## Object triggers

### About object triggers

Moab triggers are configurable actions that respond to an event occurring on a Moab object. A trigger is attached to an object and consists of both an event that may take place on the object and the action that the trigger will take.

Image 3-9: Trigger attachment



**i** Triggers are a powerful tool. Extreme caution should be taken when using them. They are useful in creating automatic responses to well-understood Moab events; however, by default triggers run as root and do exactly as they are told, meaning they require great thought and consideration to ensure that they act appropriately in response to the event.

### Use case

An administrator wants to create the following setup in Moab:

When a node's temperature exceeds 34°C, Moab reserves it. If the temperature increases to more than 40°C, Moab requeues all jobs on the node. If the node's temperature exceeds 50°C, Moab shuts it down. Moab removes the node's reservation and unsets the variables when the node cools to less than 25°C.

The administrator wants to receive an email whenever any of these events occur. All of this can be configured in Moab using triggers. To see a full example for this use case, see [Node maintenance example](#) on page 773.

### Sub content

- [About trigger variables](#) on page 775

### How-to's

- [Creating a trigger](#) on page 748
- [Using a trigger to send email](#) on page 752
- [Using a trigger to execute a script](#) on page 754
- [Using a trigger to perform internal Moab actions](#) on page 754
- [Requiring an object threshold for trigger execution](#) on page 755
- [Enabling job triggers](#) on page 755
- [Modifying a trigger](#) on page 756
- [Viewing a trigger](#) on page 757
- [Checkpointing a trigger](#) on page 757

### References

- [Job triggers](#) on page 758
- [Node triggers](#) on page 759
- [Reservation triggers](#) on page 761
- [Resource manager triggers](#) on page 762
- [Scheduler triggers](#) on page 763
- [Threshold triggers](#) on page 764
- [Trigger components](#) on page 765
- [Trigger exit codes](#) on page 773
- [Node maintenance example](#) on page 773
- [Environment creation example](#) on page 774

## How-to's

## Creating a trigger

## Context

Three methods exist for attaching a trigger to an object:

- Directly to the object via the command line
- Directly to the object via the configuration file
- As part of a template via the configuration file

`<attr>=<val>` pair delimiters, quotation marks, and other elements of the syntax may differ slightly from one method/object combination to another, but creating any trigger follows the same basic format:

```
<attr>=<val>[[{&, }<attr>=<val>] ... ]
```

The beginning of the trigger is set off by the keyword *trigger*. It is followed by a delimited list (typically by commas) of `<attr>=<val>` pairs.

Each method of trigger creation can only be used for certain Moab objects. The following table displays which objects can receive triggers via each method. The links contain examples.

Method	Objects
<b>Command line</b>	job, reservation; a trigger can be attached to any existing object using <a href="#">mschedctl -c</a>
<b>Configuration file</b>	node, reservation, RM, scheduler
<b>Template</b>	job, reservation

Triggers are composed of attributes. Only three are required for each trigger: an [EType](#) (event type), an [AType](#) (action type), and an [Action](#).

Image 3-10: Required trigger attributes

## Required trigger attributes

```
EType=start AType=exec Action="/tmp/report.pl"
```

<b>EType</b>	<b>Event Type - The event that fires the trigger</b>
<b>AType</b>	<b>Action Type - The type of action the trigger will perform</b>
<b>Action</b>	<b>Action - The actual action the trigger will perform</b>

Other attributes exist to further customize triggers. See [Trigger components on page 765](#) for more information.

### To create a Moab trigger

1. Choose an object to which, and a method by which, you will attach the trigger. Use the format and examples described in its corresponding documentation:
  - [Job triggers on page 758](#)
  - [Node triggers on page 759](#)
  - [Reservation triggers on page 761](#)
  - [Resource manager triggers on page 762](#)
  - [Scheduler triggers on page 763](#)

**i** If the trigger is to be attached to a job, you must first enable job triggers (see [Enabling job triggers on page 755](#) for more information.). Please carefully review the warning before doing so.

2. Decide whether to attach the trigger via the command line or configuration file. Verify the correct syntax.
3. Set the **EType** equal to whichever event will launch the trigger if and when it occurs on the object.

Each object has a different lifecycle, so not every event type will occur on every object. For a list of valid **ETypes** for your selected object, see the corresponding object reference page linked in step 1.

- a. To modify the timing of the trigger in any of the following ways, see [Event-modifying trigger components on page 769](#).
  - To set the trigger as rearmable and specify the amount of time the trigger must wait before firing again.
  - To set an amount of time before or after the event that the trigger will fire (See [Offset on page 769](#) for restrictions).
  - To set a specific threshold and the amount of time that the object must meet that threshold before the trigger will fire.
4. Configure the action that the trigger will take when the event happens. To do so, you must set the **AType** to a valid value for your object and specify the action. For instance, to execute a script, set the **AType** to **exec** and the **Action** to the location of the script in quotation marks. Include the name of the object on which the script will run.

```
NODECFG[node01] TRIGGER=EType=fail,AType=exec,Action="node.fail.sh node01"
```

- a. To modify the action in any of the following ways, see [Action-modifying trigger components on page 771](#).
  - To specify environment variables available to the trigger
  - To set a flag on the trigger
    - To attach any stderr output generated by the trigger to the parent object
    - To destroy the trigger if its object ends or cancels
    - To tell Moab to checkpoint the trigger
    - To set the trigger as periodic
    - To pass the object's XML information to the trigger's stdin
    - To set the trigger to reset if its object is modified
    - To set the trigger to fire under the user ID of the object's owner
  - To specify an amount of time that Moab will suspend normal operation to wait for the trigger to execute
  - To allot an amount of time that the trigger will attempt to run before it is marked as unsuccessful and the process, if any exists, is killed
  - Set a maximum number of times that a trigger will attempt to fire before it fails
- b. To give the trigger a name or description, see [Organizational trigger components on page 772](#).
- c. To configure the trigger to set or unset a variable when it fires or to require a variable to fire, see [Setting and receiving trigger variables on page 776](#).

## Creating VM triggers

### Context

Triggers can be attached to virtual machines on the command line using the `mvmctl` command.

## To create a VM trigger via the command line

1. Type the `mvmctl -m trigger=` command to modify a virtual machine. Use `trigger=` to mark the beginning of the trigger configuration.

```
> mvmctl -m trigger=
```

2. Set the `EType` to start. End with a backslash (`\`) and ampersand (`&`).

```
> mvmctl -m trigger=EType=start\&
```

3. Specify the action the trigger should take when the event occurs by setting the `AType` attribute to `changeparam`, `exec`, `internal`, `query`, or `submit`. End with a backslash and ampersand.

```
> mvmctl -m trigger=EType=start\&AType=exec\&
```

4. If the trigger launches when the job reaches a threshold, define the threshold.
5. Use the `Action` attribute to specify the action the trigger will take. Use single quotes.

```
> mvmctl -m trigger=EType=start\&AType=exec\&Action='trig.py $OID $HOSTLIST'
```

6. Configure any desired time-related attributes (The offset time, whether a trigger can fire multiple times, how often, etc.). Insert a comma between the time attribute(s) and the action.

```
> mvmctl -m trigger=EType=start\&AType=exec\&Action='trig.py $OID
$HOSTLIST',Offset=30
```

Moab launches the `trig.py` script 30 seconds after the VM starts.

7. Configure any desired variables. Verify that all attributes are separated by commas.

```
> mvmctl -m trigger=EType=start\&AType=exec\&Action='trig.py $OID
$HOSTLIST',Offset=30,sets=vmStart
```

8. Set any desired flags.
9. Submit the trigger.

## Using a trigger to send email

### Context

Mail triggers can be attached to nodes, jobs, reservations, and the scheduler. The recipient of the email depends on the object to which the trigger is attached. To select different recipient(s) and add flexibility to formatting, send email via a script [using an exec trigger](#).

### To use a trigger to send email

1. For objects that send mail to the primary user, you must configure the Moab administrator email using the [MAILPROGRAM on page 992](#) parameter.
2. Create a trigger on one of the four valid objects listed below, setting the `AType` to [mail](#) and the `Action` to the body of the message inside of quotation marks.

Object	Recipient
<b>Node</b>	The primary user (the first user listed in <a href="#">ADMINCFG[1]</a> , typically root)
<b>Job</b>	The job's owner
<b>Reservation</b>	The primary user
<b>Scheduler</b>	The primary user

- When attaching a mail trigger to all objects of a certain type, use internal variables in the **Action** to add information that is specific to an object, such as the ID, owner, time the event occurred, etc. A variable must be preceded by a dollar sign (\$).

Variable	Description
<b>\$OID</b>	Name of the object to which the trigger is attached
<b>\$OTYPE</b>	The type of object to which the trigger is attached
<b>\$TIME</b>	Time the trigger launched
<b>\$HOSTLIST</b>	Hostlist of the trigger's object (jobs and reservations)
<b>\$OWNER</b>	Owner of the trigger's object (jobs and reservations)
<b>\$USER</b>	User (jobs and reservations)

The variable is replaced with the information described above. For example, the following trigger is configured on all nodes:

```
NODECFG[DEFAULT] EType=fail,AType=mail,Action="node $OID failed at $TIME"
```

*When, for example, node node03 fails, an email is sent to the primary user with a message with the subject line "node node03 started on Sat Aug 18 11:42:00".*

## Using a trigger to execute a script

### Context

Exec triggers launch a program or script when the event occurs. A few examples of what a script might do in response to an event include:

- Execute an external program
- Send a complex email to any desired recipient(s)
- Collect diagnostics

**i** It is important to note that when a script runs via a trigger, Moab forks and performs a direct OS exec, meaning there will be no pre-processing of the command by the shell. In addition, the script runs in a new, reduced environment without the same settings and variables as the environment from which it stemmed. The script must be able to run in the reduced environment.

### To use a trigger to execute a script

1. Create or locate the script and note its location.
2. Create a trigger on the desired object, setting the **AType** to **exec** and the **Action** to location of the script or program.

```
JOBCFG[temp1] TRIGGER=EType=start,AType=exec,Offset=03:00,Action="/tmp/monitor.pl"
```

*Jobs with the temp1 template receive a trigger that executes monitor.pl three minutes after the job starts.*

## Using a trigger to perform internal Moab actions

### To perform internal actions in Moab with a trigger

- Create a trigger on a job, node, or reservation, setting the **AType** to **internal** and the **Action** to one of the following:
  - **node:-:reserve** - reserves the node to which the trigger is attached
  - **job:-:cancel** - cancels the job to which the trigger is attached
  - **reservation:-:cancel** - cancels the reservation to which the trigger is attached

The specified object reserves or cancels itself once the event occurs. See [Internal Action on page 768](#) for examples.

## Requiring an object threshold for trigger execution

### Context

Threshold triggers allow sites to configure triggers to launch based on internal scheduler statistics, such as generic metrics. For example, you might configure a trigger to warn the administrator when the percentage of nodes available is less than 25.

### To configure a threshold trigger

1. Create a trigger. Set its **EType** to *threshold*. Configure the **AType**, **Action**, and **Threshold** attributes' values based on the valid thresholds per object listed in the table found in [Threshold triggers on page 764](#).

```
NODECFG[node04] TRIGGER=EType=threshold,AType=exec,Action="$HOME/hightemp.py
$OID",Threshold=gmetric
```

2. Insert the gmetric name between brackets (such as `gmetric[temp]`). Provide a comparison operator. For valid options, see the [comparison operators table](#).
3. Provide a number or string to match against the threshold.

```
NODECFG[node04] TRIGGER=EType=threshold,AType=exec,Action="$HOME/hightemp.py
$OID",Threshold=gmetric[TEMP]>70,RearmTime=5:00
```

*Moab launches a script that warns the administrator when node04's gmetric temp exceeds 70. Moab rearms the trigger five minutes after it fires.*

## Enabling job triggers

### Context

By default common users cannot create most objects, and as a result, common users also cannot create triggers. The exception, however, is jobs. Because common users can create jobs and triggers generally run as root, additional security is necessary to ensure that not all users can create triggers. For this reason, job triggers are disabled by default.



Because triggers generally run as root, any user given the power to attach triggers has the power to run scripts and commands as root. It is recommended that you only enable job triggers on closed systems in which human users do not have access to directly submit jobs.

To give specific users permission to create job triggers, you must create a QoS, set the *trigger* flag, and add users to it.

### To enable job triggers

1. In the `moab.cfg` file, create a QoS and set the *trigger* flag.

```
QOSCFG[triggerok] QFLAGS=trigger
```

2. Add users to the QoS who should be allowed to add triggers to jobs.

```
USERCFG[joe] QDEF=triggerok
```

User `joe` is added to the *triggerok* QoS, giving him both the power to create job triggers and root access to the machine.

## Modifying a trigger

### Context

You can modify a trigger at any time by updating its settings in the Moab configuration file (`moab.cfg`). This will update most triggers at the beginning of the next Moab iteration; however, modifying template triggers (configured using [RSVPROFILE](#) or [JOBFCFG](#)) will not update the instances of the trigger that were attached to individual reservations or jobs on creation. The modification will only affect the triggers that the template attaches to future objects.

Any trigger with a specified name can be modified using the `mschedctl -m` command in the following format:

```
mschedctl -m trigger: <triggerID><attr1>=<val1><attr2>=<val2>
```



Modifying triggers on the command line does not change their configuration in `moab.cfg`. Except for reservations that are checkpointed, changes made dynamically are lost when Moab restarts.

For example, the procedure below demonstrates how to modify the following trigger so that the offset is 10 minutes instead of 5 and so that Moab will attempt to fire the trigger up to 10 times if it fails. Assume your trigger currently looks like this:

```
NODECFG[DEFAULT] EType=fail,AType=exec,Action="/scripts/node_fail.pl",Name=nodeFailTrig,Offset=00:05:00,MultiFire=TRUE,RearmTime=01:00:00
```

### To modify a trigger

1. Type `mschedctl -m` into the command line and set off the trigger modification with `trigger:<id>`. Use the trigger's assigned ID or specified name to state which trigger will receive the modification.

```
> mschedctl -m trigger:nodeFailTrig
```

2. Type any changing attributes equal to the new value. Separate multiple modifications with a space between each `<attr>=<val>` pair. In this case, set the **Offset** and **MaxRetry** attributes the following way:

```
> mschedctl -m trigger:nodeFailTrig Offset=00:10:00 MaxRetry=10
```

The newly-specified attributes replace the original ones. Trigger `nodeFailTrig` now has an offset of 10 minutes and will try to fire a maximum of 10 times if it fails. The new trigger has the following attributes:

```
EType=fail,AType=exec,Action="/scripts/start_rsv.pl",Name=nodeFailTrig,Offset=00:10:00,MultiFire=TRUE,RearmTime=01:00:00,MaxRetry=10
```

## Viewing a trigger

### Context

Moab provides a list of triggers when you run the [mddiag -T](#) command. You can view a specific trigger by running `mddiag -T` in the following format:

```
mddiag -T [<triggerID>|<objectID>|<triggerName>|<objectType>]
```

### To view a trigger

1. Type `mddiag -T` in the command line.
2. Specify either the trigger ID, the trigger name, the name of the object to which the trigger is attached, or the type of object to which the trigger is attached. For example, if you wanted to view information about a trigger with ID `trigger.34` and name `jobFailTrigger`, which is attached to job `job.493`, you could run any of the following commands:

```
> mdiag -T trigger.34
> mdiag -T job.493
> mdiag -T jobFailTrigger
> mdiag -T job
```

*The output of the first command would provide basic information about `trigger.34`; the second command, information about all triggers attached to `job.493` that the user can access; the third command, basic information about `jobFailTrigger`; and the fourth command, basic information about all triggers attached to jobs that the user can access.*

3. Optional: to view additional information about the trigger, run the same command with the `-v` flag specified after `-T`.

```
> mdiag -T -v job.493
```

This mode outputs information in multiple lines.

4. Optional: to view detailed information about all triggers available to you, use the `mddiag -T -v` command. This outputs all triggers available to the user in a single line for each trigger. It provides additional state information about triggers, including reasons triggers are currently blocked.

```
> mdiag -T -v
```

## Checkpointing a trigger

### Context

Checkpointing is the process of saving state information when Moab is shut down. In general, triggers defined in the `moab.cfg` file are not checkpointed but are recreated when Moab starts. The exception is the [JOB\\_CFG](#) parameter, which attaches triggers to jobs as they are created. There are two cases in which you may want to tell Moab to checkpoint a trigger:

- If a trigger is defined in the `moab.cfg` file but was created at the command line
- When creating a trigger using the [mschedctl on page 264](#) command

## To checkpoint a trigger

1. Locate the trigger to be checkpointed in the `moab.cfg` file, create one on the command line, or modify a trigger dynamically (See [Modifying a trigger on page 756](#) for more information). Attach the *checkpoint* flag using the **FLAGS** attribute. For more information about flags, see [Flags on page 771](#).

```
FLAGS=checkpoint
```

2. If you are working in the configuration file, save the changes. Moab will now checkpoint your trigger.

## References

### Job triggers

For security reasons, job triggers are disabled by default. They must be enabled in order to successfully attach triggers to jobs (See [Enabling job triggers on page 755](#) for more information.).

Triggers attached to jobs follow the same basic rules and formats as attaching them to other objects; however, not all attribute options are valid for each object. Jobs, like other objects, have a unique set of trigger rules. The table below details the methods, options, and other notable details associated with attaching triggers to jobs.

### Creation methods

Method	Format	Example
<b>Command line on job creation:</b> <a href="#">msub -l</a>	<code>msub &lt;jobName&gt; -l 'trig=&lt;trigSpec&gt;'</code> Attributes are delimited by backslash ampersand (\&).	<pre style="border: 1px dashed gray; padding: 5px;">&gt; msub my.job -l 'trig=EType=create\&amp;AType=exec\&amp;Action="/jobs/my_trigger.pl"\&amp;Offset=10:00'</pre>
<b>Command line on existing job:</b> <a href="#">mschedctl -c</a>	<code>mschedctl -c trigger &lt;trigSpec&gt; -o job:&lt;jobID&gt;</code>	<pre style="border: 1px dashed gray; padding: 5px;">&gt; mschedctl -c trigger EType=end,AType=mail,Action="Job moab.54 has ended" -o job:moab.54</pre>
<b>Job template in moab.cfg:</b> <a href="#">JOBCFG</a>	<code>JOBCFG[&lt;templateName&gt;] TRIGGER=&lt;trigSpec&gt;</code>	<pre style="border: 1px dashed gray; padding: 5px;">JOBCFG[vmcreate] TRIGGER=,EType=end,AType=exec,Action="/tmp/jobEnd.sh"</pre>

## Valid event types

- [cancel on page 768](#)
- [checkpoint on page 768](#)
- [create on page 768](#)
- [end on page 768](#)
- [hold on page 769](#)
- [modify on page 769](#)
- [preempt on page 769](#)
- [start on page 769](#)

## Valid action types

- [changeparam](#)
- [exec](#)
- [internal](#)
- [mail](#)

## Mail recipient

The job's owner

See [Using a trigger to send email on page 752](#) for more information.

## Node triggers

Triggers attached to nodes follow the same basic rules and formats as attaching them to other objects; however, not all attribute options are valid for each object. Nodes, like the other objects, have a unique set of trigger rules. The table below details the methods, options, and other notable details that come with attaching triggers to nodes.

### Creation methods

Method	Format	Example
<b>Command line on existing node:</b> <a href="#">mschedctl - c</a>	<code>mschedctl -c trigger &lt;trigSpec&gt; -o node:&lt;nodeID&gt;</code>	<pre>&gt; mschedctl -c trigger EType=fail, AType=exec, Action="/tmp/nodeFailure.sh" -o node:node01</pre>

Method	Format	Example
<b>Node configuration in moab.cfg:</b> <a href="#">NODECFG</a>	NODECFG [<name>] TRIGGER= <trigSpec>	<pre>NODECFG[node04] TRIGGER=EType=threshold,AType=exec,Action="\$HOME/hightemp.py \$OID",Threshold=gmetric[TEMP]&gt;70</pre>

## Valid event types

- [create](#) on page 768
- [discover](#) on page 768
- [end](#) on page 768
- [fail](#) on page 769
- [standing](#) on page 769
- [threshold](#) on page 769

## Valid action types

- [changeparam](#)
- [exec](#)
- [internal](#)
- [mail](#)

## Thresholds

Node threshold settings	
Valid ETypes	threshold
Valid Threshold types	gmetric

## Mail recipient

The user listed first in [ADMINCFG\[1\]](#) (usually root)

See [Using a trigger to send email](#) on page 752 for more information.

## Reservation triggers

Triggers attached to reservations follow the same basic rules and formats as attaching them to other objects; however, not all attribute options are valid for each object. Reservations, like the other objects, have a unique set of trigger rules. The table below details the methods, options, and other notable details that come with attaching triggers to reservations.

### Creation methods

Method	Format	Example
<b>Command line on reservation creation:</b> <a href="#">mrsvctl -T</a>	<code>mrsvctl -c -h &lt;host-list&gt; -T &lt;trigSpec&gt;</code>	<pre>&gt; mrsvctl -c -h node01 -T EType=start,AType=exec, Action="/scripts/node_start.pl"</pre>
<b>Command line on existing reservation:</b> <a href="#">mschedctl -c</a>	<code>mschedctl -c trigger &lt;trigSpec&gt; -o rsv:&lt;rsvID&gt;</code>	<pre>&gt; mschedctl -c trigger EType=modify,AType=mail,Action="Reservation system.4 has been modified" -o rsv:system.4</pre>
<b>Standing reservation configuration in moab.cfg:</b> <a href="#">SRCFG</a>	<code>SRCFG [&lt;name&gt;] TRIGGER= &lt;trigSpec&gt;</code>	<pre>SRCFG[Mail2] TRIGGER=EType=start,Offset=200,AType=exec,Action="/tmp/email.sh"</pre>
<b>Reservation template in moab.cfg:</b> <a href="#">RSVPROFILE</a>	<code>RSVPROFILE [&lt;name&gt;] TRIGGER= &lt;trigSpec&gt;</code>	<pre>RSVPROFILE[rsvtest] TRIGGER=EType=cancel,AType=exec,Action="\$HOME/logdate.pl TEST CANCEL \$VPCHOSTLIST \$OID \$HOSTLIST \$ACTIVE"</pre>

### Valid event types

- [create on page 768](#)
- [end on page 768](#)
- [modify on page 769](#)
- [standing on page 769](#)
- [start on page 769](#)
- [threshold on page 769](#)

## Valid action types

- [cancel](#)
- [changeparam](#)
- [exec](#)
- [internal](#)
- [jobpreempt](#)
- [mail](#)

## Thresholds

Node threshold settings	
Valid ETypes	threshold
Valid Threshold types	usage

## Mail recipient

The owner of the reservation. If the owner is unknown or not a user, the first user listed first in [ADMINCFG](#) (usually `root`).

See [Using a trigger to send email on page 752](#) for more information.

## Resource manager triggers

Triggers attached to the resource manager follow the same basic rules and formats as attaching them to other objects; however, not all attribute options are valid for each object. The resource manager, like other objects, has a unique set of trigger rules. The table below details the methods, options, and other notable details that come with attaching triggers to RMs.

## Creation methods

Method	Format	Example
<b>Command line on existing RM:</b> <a href="#">mschedctl</a> <b>-c</b>	<code>mschedctl -c trigger</code> <code>&lt;trigSpec&gt; -o rm:&lt;rmID&gt;</code>	<pre>&gt; mschedctl -c trigger EType=start,AType=exec,Action="/tmp/rmStart.sh" -o rm:torque</pre>

Method	Format	Example
<b>RM configuration in moab.cfg:</b> <a href="#">RMCFG</a>	RMCFG [<name>] TRIGGER= <trigSpec>	<pre>RMCFG[base] TRIGGER=EType=fail,AType=exec,Action="/opt/moab/tools/diagnose_rm.pl \$OID"</pre>

## Valid event types

- [fail](#) on page 769
- [threshold](#) on page 769

## Valid action types

- [changeparam](#)
- [exec](#)
- [internal](#)

## Scheduler triggers

Triggers attached to the scheduler follow the same basic rules and formats as attaching them to other objects; however, not all attribute options are valid for each object. The scheduler, like the other objects, has a unique set of trigger rules. The table below details the methods, options, and other notable details associated with attaching triggers to the scheduler.

## Creation methods

Method	Format	Example
<b>Command line on existing scheduler:</b> <a href="#">mschedctl -c</a>	mschedctl -c trigger <trigSpec> -o sched:<schedID>	<pre>&gt; mschedctl -c trigger EType=end,AType=exec,Action="/tmp/startRsvs.sh" -o sched:moab</pre>
<b>Scheduler configuration in moab.cfg:</b> <a href="#">SCHEDCFG</a>	SCHEDCFG [<name>] TRIGGER=<trigSpec>	<pre>SCHEDCFG[MyCluster] TRIGGER=EType=fail,AType=mail,Action="scheduler failure detected on \$TIME",RearmTime=15:00</pre>

## Valid event types

- [create on page 768](#)
- [end on page 768](#)
- [fail on page 769](#)
- [modify on page 769](#)
- [standing on page 769](#)
- [start on page 769](#)

## Valid action types

- [changeparam](#)
- [exec](#)
- [internal](#)
- [mail](#)

## Mail recipient

The user listed first in [ADMINCFG](#) (usually `root`)

See [Using a trigger to send email on page 752](#) for more information.

## Threshold triggers

The following table identifies the object event, and usage types with which the threshold event/action type feature works.

Object type	Event Type	Usage types
<b>Node</b>	Threshold	gmetric
<b>Reservation</b>	Threshold	usage

The following table defines each of the usage types:

Usage type	Description
<b>gmetric</b>	Generic performance metrics configured in Moab (See Enabling Generic Metrics for more information).
<b>usage</b>	The percentage of the resource being used (not idle).

The following table defines each of the threshold trigger comparison operators:

Comparison operator	Value
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
==	Equal to

## Examples

*Example 3-150: Reservation usage threshold*

```
SRCFG[res1] TRIGGER=EType=threshold,AType=mail,Action="More than 75% of reservation
res1 is being used",Threshold=usage>75,FailOffset=1:00
```

*When more than 75% of the reservation has been in use for at least a minute, Moab fires a trigger to notify the primary user.*

## Trigger components

### Required trigger components

#### AType

Action type	Description
<b>cancel</b>	Cancels the object
<b>changeparam</b>	Causes Moab to give a parameter to a new value
<b>exec</b>	Launches an external program or script on the command line when the dependencies are fulfilled. See <a href="#">Using a trigger to execute a script on page 754</a> for more information.
<b>internal</b>	Modifies Moab without using the command line. See <a href="#">Using a trigger to perform internal Moab actions on page 754</a> for more information.

Action type	Description
<b>jobpreempt</b>	Indicates the preempt policy to apply to all jobs currently allocated resources assigned to the trigger's parent reservation
<b>mail</b>	Causes Moab to send mail. See <a href="#">Using a trigger to send email on page 752</a> for more information.

## Action

Cancel Action	
<b>Format</b>	NONE
<b>Description</b>	Indicates that Moab should cancel the reservation when the event occurs. No action should be specified.
<b>Example</b>	<pre>Etype=threshold,Threshold=usage&lt;10,FailOffset=1:00,AType=cancel</pre> <p><i>When less than 10% of the reservation has been in use for a minute, Moab cancels it.</i></p>

Changeparam Action	
<b>Format</b>	Action="<STRING>"
<b>Description</b>	Specifies the parameter to change and its new value (using the same syntax and behavior as the <a href="#">changeparam on page 363</a> command)
<b>Example</b>	<pre>Atype=changeparam,Action="JOBPCURGETIME 02:00:00"</pre> <p><i>Moab maintains detailed job information for two hours after a job has completed.</i></p>

Jobpreempt Action	
<b>Format</b>	Action="cancel checkpoint requeue suspend"
<b>Description</b>	Signifies <a href="#">PREEMTPOLICY</a> to apply to jobs that are running on allocated resources

Jobpreempt Action	
<b>Example</b>	<pre>RSVPROFILE [adm1] TRIGGER=EType=start,Offset=-240,AType=jobpreempt,Action="cancel"</pre> <p><i>40 minutes after the reservation adm1 starts, all jobs using the reservation's resources adopt a <b>PREEMTPOLICY</b> of <b>cancel</b>.</i></p>

Mail Action	
<b>Format</b>	Action="<MESSAGE>"
<b>Description</b>	<p>When <b>AType=mail</b>, the <b>Action</b> parameter contains the message body of the email. This can be configured to include certain variables. See <a href="#">Using a trigger to send email on page 752</a> for details.</p> <p>Mail triggers can be configured to launch for node failures, reservation creation or release, scheduler failures, and even job events. In this way, site administrators can keep track of scheduler events through email.</p> <p>The email comes from <i>moabadmin</i>, has a subject of <i>moab update</i>, and has a body of whatever you specified in the <b>Action</b> attribute. The recipient list depends on the type of object the trigger is attached to.</p> <ul style="list-style-type: none"> <li>• <b>Node</b> - The primary user (first listed in <a href="#">ADMINCFG[1]</a>), typically <i>root</i></li> <li>• <b>Scheduler</b> - The primary user</li> <li>• <b>Job</b> - The user who owns the job</li> <li>• <b>Reservation</b> - The primary user</li> </ul>
<b>Example</b>	<pre>NODECFG[DEFAULT] TRIGGER=EType=fail,AType=mail,Action="node \$OID will failed.",Offset=05:00:00</pre> <p><i>This example sends an email to the primary administrator informing him/her that the node (including the node ID) has failed.</i></p>

Exec Action	
<b>Format</b>	Action="<script>"
<b>Description</b>	<p>Exec triggers will launch an external program or script when their dependencies are fulfilled. The following example will submit <i>job.cmd</i> and then execute <i>monitor.pl</i> three minutes after the job is started. See <a href="#">Using a trigger to execute a script on page 754</a> for more information.</p>
<b>Example</b>	<pre>&gt; msub -l trig=EType=start\&amp;AType=exec\&amp;Action="/tmp/monitor.pl" job.cmd\&amp;Offset=03:00</pre>

Internal Action	
<b>Format</b>	Action=" <i>&lt;objectType&gt;</i> ::<cancel reserve>"
<b>Description</b>	<p>A couple different actions are valid depending on what type of object the internal trigger is acting upon. The following list shows the available actions:</p> <ul style="list-style-type: none"> <li>• Reserve a node</li> <li>• Cancel a job</li> <li>• Cancel a reservation</li> </ul> <p>See <a href="#">Using a trigger to perform internal Moab actions on page 754</a> for more information.</p>
<b>Example</b>	<pre>NODECFG[node01] TRIGGER=EType=start,AType=internal,Action="node::-reserve"</pre> <p><i>When node01 starts, it becomes a reservation.</i></p> <pre>&gt; msub moab.3 -l 'trig=EType=fail\&amp;AType=internal\&amp;Action="job::-cancel"</pre> <p><i>If moab.3 fails, Moab cancels it.</i></p> <pre>&gt; mrsvctl -c -a user==joe -h node50 -T EType=start,AType=internal,Action="reservation::-cancel",Offset=10:00</pre> <p><i>User joe's jobs are given a ten-minute window to start, then the reservation cancels.</i></p>

## EType

Event type	Description
<b>cancel</b>	The event is triggered when the parent object is either canceled or deleted.
<b>checkpoint</b>	Triggers fire when the job is checkpointed. <i>checkpoint</i> triggers can only be attached to jobs.
<b>create</b>	Triggers fire when the parent object is created. <i>create</i> triggers can be attached to nodes, jobs, reservations, and the scheduler (when attached to the scheduler, triggers fire when Moab starts).
<b>discover</b>	Triggers fire when the node is loaded from a resource manager and Moab cannot recognize it nor find it in the checkpoint file.
<b>end</b>	Triggers fire when the parent object ends. <i>end</i> triggers can be attached to nodes, jobs, reservations, and the scheduler (when attached to the scheduler, triggers fire when Moab shuts down).

Event type	Description
<b>fail</b>	Triggers fire when the resource manager is in a corrupt or down state for longer than the configured fail time, or when Moab detects a corruption in a node's reservation table. <i>fail</i> triggers can be attached to jobs, nodes, resource managers, and the scheduler.
<b>hold</b>	Triggers fire when the job is put on hold. <i>hold</i> triggers can only be attached to jobs.
<b>modify</b>	Triggers fire when the parent object is modified. <i>modify</i> triggers can be attached to jobs and reservations
<b>preempt</b>	Triggers fire when the job is preempted. <i>preempt</i> triggers can only be attached to jobs.
<b>standing</b>	Triggers fire multiple times based on a certain period. They can be used with <a href="#">Period</a> and <a href="#">Offset</a> attributes. <i>standing</i> triggers can be attached to nodes and the scheduler.
<b>start</b>	Triggers fire when the parent object or Moab starts. <i>start</i> triggers can be attached to jobs, reservations, resource managers, and the scheduler (when Moab starts and at the beginning of Moab's first iteration).
<b>threshold</b>	Triggers fire when a threshold, such as usage or a gmetric comparison, is true. <i>threshold</i> triggers can be attached to nodes and reservations. Triggers with <b>ETypes</b> set to <i>threshold</i> must include the <a href="#">Threshold</a> attribute.

## Event-modifying trigger components

The following trigger attributes modify the event that causes the trigger to fire.

RearmTime	
<b>Possible Values</b>	[[HH:]MM:]SS
<b>Description</b>	The amount of time that must pass before a trigger can fire again. <b>RearmTime</b> is enforced from the trigger event time.
<b>Usage Notes</b>	---

Offset	
<b>Possible Values</b>	[-] [[HH:]MM:]SS

Offset	
<b>Description</b>	The relative time offset from event when trigger can fire
<b>Usage Notes</b>	<ul style="list-style-type: none"> <li>• Only end triggers can have a negative value for <b>Offset</b>.</li> <li>• <b>Offset</b> cannot be used with cancel.</li> </ul>

Period	
<b>Possible Values</b>	<i>Minute, Hour, Day, Week, Month, Infinity</i>
<b>Description</b>	The period at which the trigger will regularly fire
<b>Usage Notes</b>	---

Threshold	
<b>Possible Values</b>	Threshold={<metric>[<metricName>]}{> >= < <= ==}<FLOAT> Where <metric> is one of: <ul style="list-style-type: none"> <li>• <b>gmetric</b></li> <li>• <b>usage</b></li> </ul>
<b>Description</b>	When the object meets, drops below, or increases past the configured <b>Threshold</b> , the trigger will fire.
<b>Usage Notes</b>	Threshold triggers allow sites to configure triggers to launch based on internal scheduler statistics, such as the usage of a reservation.

FailOffset	
<b>Possible Values</b>	[ [HH:]MM:]SS
<b>Description</b>	The time that the threshold condition must exist before the trigger fires
<b>Usage Notes</b>	Use with fail triggers to avoid transient triggers.

## Action-modifying trigger components

Flags	
<b>Possible Values</b>	<p>Flags=&lt;flag&gt;[:&lt;flag&gt;] or Flags=[&lt;flag&gt;][[&lt;flag&gt;]]</p> <p><i>attacherror</i> - If the trigger outputs anything to stderr, Moab attaches it as a message to the trigger object.</p> <p><i>cleanup</i> - If the trigger is still running when the parent object completes or is canceled, Moab kills the trigger.</p> <p><i>checkpoint</i> - Moab always checkpoints this trigger. For more information, see <a href="#">Checkpointing a trigger on page 757</a>.</p> <p><i>objectxmlstdin</i> - Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.</p> <p><i>resetonmodify</i> - The trigger resets if its object is modified, even if <b>RearmTime</b> is not set.</p> <p><i>user</i> - The trigger executes under the user ID of the object's owner. If the parent object is the scheduler, you may explicitly specify the user using the format <code>user+&lt;username&gt;</code>. For example: <code>Flags=user+john</code>.</p>
<b>Description</b>	Specifies various trigger behaviors and actions
<b>Usage Notes</b>	<p>When specifying multiple flags, each flag can be delimited by colons (:) or with square brackets; for example:</p> <p>Flags=[user] [cleanup] or Flags=user:cleanup</p>

BlockTime	
<b>Possible Values</b>	[[HH:]MM:]SS
<b>Description</b>	The amount of time Moab will suspend normal operation to wait for trigger execution to finish
<b>Usage Notes</b>	Use caution; Moab will completely stop normal operation until <b>BlockTime</b> expires.

ExpireTime	
<b>Possible Values</b>	<INTEGER>
<b>Description</b>	The time at which trigger should be terminated if it has not already been activated
<b>Usage Notes</b>	---

Timeout	
<b>Possible Values</b>	[+ -] [[HH:]MM:]SS
<b>Description</b>	The time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed
<b>Usage Notes</b>	---

MaxRetry	
<b>Possible Values</b>	MaxRetry=<INTEGER>
<b>Description</b>	The number of times <b>Action</b> will be attempted before the trigger is designated a failure
<b>Usage Notes</b>	If <b>Action</b> fails, the trigger will restart immediately (up to <b>MaxRetry</b> times). If it fails more than <b>MaxRetry</b> times, the trigger has failed. This restart ignores <b>FailOffset</b> and <b>RearmTime</b> .

## Organizational trigger components

Name	
<b>Possible Values</b>	Name=<STRING>
<b>Description</b>	Name of the trigger
<b>Usage Notes</b>	Because Moab uses its own internal ID to distinguish triggers, the <b>Name</b> need not be unique. Only the first 16 characters of <b>Name</b> are stored by Moab.

Description	
<b>Possible Values</b>	Description=<STRING>
<b>Description</b>	Description of the trigger
<b>Usage Notes</b>	---

## Trigger exit codes

By default Moab considers any non-zero exit code as a failure and marks the trigger as having failed. If a trigger is killed by a signal outside of Moab, Moab treats the signal as the exit code and (in almost all cases) marks the trigger as having failed. Only exec triggers that exit with an exit code of 0 are marked as successful.

## Node maintenance example

### Example scenario

An administrator wants to create the following setup in Moab:

When a node's temperature exceeds 34°C, Moab reserves it. If the temperature increases to more than 40°C, Moab requeues all jobs on the node. If the node's temperature exceeds 50°C, Moab shuts it down. Moab removes the node's reservation and unsets the variables when the node cools to less than 25°C. The administrator wants to receive an email whenever any of these events occur.

The first trigger reserves the node when its reported temperature exceeds 34°C. Note that the gmetric name in the trigger must match the name of the configured gmetric exactly, including its case (See [Enabling Generic Metrics on page 575](#) for more information.).

```
NODECFG[DEFAULT] TRIGGER=Description="ThresholdA",EType=threshold,Threshold=gmetric
[temp]>34,AType=internal,Action="node::-reserve",RearmTime=30,Offset=2:00,Sets=temp_
rsv
```

The administrator wants the trigger to fire any time a node overheats, so it must be rearmable. It also needs to specify that the node must be over 34°C for at least two minutes for Moab to reserve it. If the trigger succeeds, it will set a variable to be received by the next trigger in order to make them sequential.

The administrator wants to know when this trigger has fired, so another trigger will send an email once the first trigger has fired and the `temp_rsv` variable is set. This one does so via a script:

```
NODECFG[DEFAULT] Trigger=Description="Email on
Reservation",EType=start,AType=exec,Action="$TOOLSDDIR/node_temp_emailReserve.pl
$OID",RearmTime=3:00,Requires=temp_rsv
```

The second threshold trigger requeues the node's jobs if the node exceeds 40°C and the `temp_rsv` variable is set. It uses a script to do so. It sets `node_evac` variable when it fires, regardless of whether it succeeds or fails.

```
NODECFG[DEFAULT] Trigger=Description="Threshold B",EType=threshold,Threshold=gmetric
[temp]>40,AType=exec,Action="$TOOLSDDIR/node_evacuate.pl
$OID",RearmTime=3:00,requires=temp_rsv,Sets=node_evac,!node_evac
```

The administrator wants another email to inform him that the node is still overheating and has been evacuated. Another email trigger fires once it receives the `node_evac` variable.

```
NODECFG[DEFAULT] Trigger=Description="Email on
Evacuation",EType=start,AType=exec,Action="$TOOLSDDIR/node_temp_emailEvac.pl
$OID",RearmTime=3:00,Requires=node_evac
```

The third threshold trigger uses a script to shut down the node if the temp gmetric exceeds 50 and the node\_evac variable is set. It sets a node\_shutdown variable to be received by the notification email.

```
NODECFG[DEFAULT TRIGGER=Description="Threshold C",EType=threshold,Threshold=gmetric
[temp]>50,AType=exec,Action="$TOOLSDIR/node_shutdown.pl
$OID",RearmTime=3:00,Requires=node_evac,Sets=node_shutdown

NODECFG[DEFAULT] Trigger=Description="Email on
Shutdown",EType=start,AType=exec,Action="$TOOLSDIR/node_temp_emailShutdown.pl
$OID",RearmTime=3:00,Requires=node_shutdown
```

The final trigger removes the reservation and unsets the variables once the node's temp gmetric is less than 25.

```
NODECFG[DEFAULT] Trigger=Description="Remove
Reservation",EType=threshold,Threshold=gmetric[temp]
<25,AType=exec,Action="opt/moab/bin/mrsvctl -r r:$OID",RearmTime=3:00,Requires=temp_
rsv,unsets=temp_rsv.node_evac.node_shutdown
```

## Environment creation example

### Example scenario

An administrator wants to create the following setup in Moab:

If a user requests an environment, she must have the permission of her two managers and the administrator. If all three approve, then the environment builds. The user is sent email messages informing her of the environment's end date in case she would like an extension. These are sent 7, 3, and 1 days prior to the environment's ending.

The administrator wants to require his and the managers' approval of any modifications the user makes to her environment so that it cannot be extended without consent.

The first trigger requests manager and administrator approval in response to the user's environment request. So in the event of a reservation's creation, a script is used to send messages to the administrator and manager. The internal variable OWNER is used to indicate to the recipients (via the script) which user is requesting the environment.

```
RSVPROFILE[envSetup] TRIGGER=EType=create,AType=exec,Action="envRequest.sh $OWNER"
```

The managers and administrator use an external program to approve or reject the request. On approval, a variable is sent back to Moab (to the reservation specifically). Once all three variables are set, the environment can start. In this example, the variables are called approval1, approval2, and approval3.

```
RSVPROFILE[envSetup]
TRIGGER=EType=start,AType=exec,Action="buildScript",Requires=approval1.approval2.appro
val3
```

As it is configured now, the reservation will continue to reserve the requested resources regardless of whether all three approvals are given. So, in case approval is not given, the next trigger cancels the reservation 7 days after its creation if the three variables are not set.

```
RSVPROFILE [envSetup]
TRIGGER=EType=create,Offset=7:00:00,AType=internal,Action="rsv::-cancel",Requires=!approval1.!approval2.!approval3
```

Every remaining trigger in this series is meant to fire for an approved environment and must require the approval variables. Otherwise these notifications would be sent to users who do not have the environment they requested. The next triggers must be rearmable so that it can fire again if necessary; however, they should be set to just over the amount of time left on the reservation so that it doesn't fire again for the same environment. The notification triggers use the **Offset** attribute to fire at the administrator's requested times (7, 3, and 1 day(s) prior to the environment's end).

```
RSVPROFILE [envSetup] TRIGGER=EType=end,Offset=7:00:00,AType=exec,Action="weekNotification.sh",RearmTime=7:00:00:02,Requires=approval1.approval2.approval3

RSVPROFILE [envSetup] TRIGGER=EType=end,Offset=3:00:00,AType=exec,Action="3dayNotification.sh",RearmTime=3:00:00:02,Requires=approval1.approval2.approval3

RSVPROFILE [envSetup] TRIGGER=EType=end,Offset=1:00:00,AType=exec,Action="dayNotification.sh",RearmTime=1:00:00:02,Requires=approval1.approval2.approval3
```

The next trigger requests administrator and manager approval when the environment is modified. The problem is that the trigger must be rearmable in case of multiple modifications and each time the [RearmTime on page 769](#) is reached, Moab will fire the trigger based on the *first* instance of modification. To resolve this issue, this modification trigger requires a *modify* variable. When the reservation is modified, the *modify* variable is set.

```
RSVPROFILE [envSetup]
TRIGGER=EType=modify,AType=exec,Action="modify.sh",RearmTime=1:00:00,Requires=approval1.approval2.approval3.!modify,Sets=modify
RSVPROFILE [envSetup]
TRIGGER=EType=modify,AType=exec,Action="modificationRequest.sh",RearmTime=5:00,Requires=approval1.approval2.approval3.modify,Unsets=modify
```

The final triggers notify the user of the end of the environment.

```
RSVPROFILE [envSetup]
TRIGGER=EType=end,AType=exec,Action="end.sh",Requires=approval1.approval2.approval3
```

The same trigger is repeated for the **cancelEType** in case the environment ends unexpectedly.

```
RSVPROFILE [envSetup]
TRIGGER=EType=cancel,AType=exec,Action="end.sh",Requires=approval1.approval2.approval3
```

## Trigger variables

### About trigger variables

Trigger variables are pieces of information that pass from trigger to trigger. They allow triggers to fire based on another trigger's behavior, state, and/or output. A variable can be a required condition for a trigger to fire; for instance, a trigger might be set to launch when a reservation starts, but only if it has received a variable from another trigger indicating that a specific node has started first. Variables give

greater flexibility and power to a site administrator who wants to automate certain tasks and system behaviors.

Variables can be used to define under what circumstances the trigger will fire. Many Moab objects have their own variables and each object's variable name space is unique. Triggers can use their own variables or the variables attached to their parent objects. A trigger's variable name space is limited to itself and its parent object. Variables do not have to be unique across all objects.

#### How-to's

- [Setting and receiving trigger variables on page 776](#)
- [Externally injecting variables into job triggers on page 777](#)
- [Exporting variables to parent objects on page 777](#)
- [Requiring variables from generations of parent objects on page 778](#)
- [Requesting name space variables on page 778](#)

#### References

- [Dependency trigger components on page 779](#)
- [Internal variables on page 780](#)

## How-to's

### Setting and receiving trigger variables

#### Context

Following is an example of how comparative dependencies can be expressed when creating a trigger.

#### To set and require variables

1. Create a trigger.

```
EType=start,AType=exec,Action="/tmp/trigger1.sh"
```

2. Use the **Sets** attribute to set a variable if the trigger succeeds. You can precede the variable with "!" to indicate that the variable should be set if the trigger fails. You can specify more than one variable by separating them with a period.

```
AType=exec,Action="/tmp/trigger1.sh",EType=start,Sets=!Var1.Var2
```

*The trigger sets variable Var2 when it succeeds and variable Var1 when it fails.*

3. Set up the recipient trigger(s). Use the **Requires** attribute to receive the variable(s). Note that preceding the variable with "!" means that the variable must not be set in order for the trigger to fire.

```
AType=exec,Action="/tmp/trigger1.sh",EType=start,Sets=!Var1.Var2
AType=exec,Action="/tmp/trigger2.sh",EType=start,Requires=Var1
AType=exec,Action="/tmp/trigger3.sh",EType=start,Requires=Var2
```

*The second trigger will launch if Var1 has been set (the first trigger failed), and the third trigger will launch if Var2 is set (the first trigger succeeded).*

## 4. Refine the requirement with comparisons.

## a. Use the following format:

```
<varID>[:<type>[:<varVal>]]
```

## b. Change &lt;varID&gt; to the variable name.

c. Use any of the comparisons found on the [Trigger variable comparison types on page 779](#) page in place of <type>:

## d. Set the value that the variable will be compared against.

```
AType=exec,Action="/tmp/trigger2.sh",EType=start,Requires=Var1:eq:45
AType=exec,Action="/tmp/trigger3.sh",EType=start,Requires=Var2:ne:failure1
```

*The first trigger fires if Var1 exists and has a value of 45. The second trigger fires if Var2 does not have a string value of failure1.*

## Externally injecting variables into job triggers

### Context

Job triggers are able to see the variables in the job object to which it is attached. This means that, for triggers that are attached to job objects, another method for supplying variables exists. Updating the job object's variables effectively updates the variable for the trigger.

### To externally inject variables into job triggers

Use the `mjobctl -m` command to set a variable to attach to a job.

```
> mjobctl -m var=Flag1=TRUE 1664
```

The variable `Flag1` is set. This will be available to any trigger attached to job 1664.

## Exporting variables to parent objects

### To export variables to parent objects

1. When setting a variable, indicate that the variable is to be exported to the parent object by using a caret (^).

```
AType=exec,Action="/tmp/trigger1.sh",EType=start,Sets=Var1.^Var2
AType=exec,Action="/tmp/trigger2.sh",EType=start,Requires=Var1
AType=exec,Action="/tmp/trigger3.sh",EType=start,Requires=Var2
```

`Var2` is exported to the parent object if the trigger fails. It can be used by job and reservation triggers at the same level or by parent objects.

2. Optional: if running a script, you can set a variable as a string to pass up to the parent object.

a. Set the variable to pass up to the parent object with the caret (^). Use the `exec AType` to run a script.

```
AType=exec,Action="/tmp/trigger.sh",EType=start,Sets=^Var1
```

The trigger sets `Var1` when it completes successfully. Because the trigger launches a script, a string value can be set for `Var1`.

- b. Declare the variable's string value on its own line in the trigger stdout.

```
EXITCODE=15
Var1=linux
```

*Var1 has the value of linux and is passed up to the parent object. This is useful in workflows in which a trigger may depend on the value given by a previous trigger.*

**i** To return multiple variables, simply print out one per line.

## Requiring variables from generations of parent objects

### Context

By default, triggers look for variables to fulfill dependencies in the object to which they are directly attached. If they are attached to a job object, they will also look in the job group, if defined. However, it is not uncommon for objects to have multiple generations of parent objects. If the desired behavior is to search through all parent objects, do the following task.

### To require variables from generations of parent objects

- Set the **Requires** attribute in the trigger to the required variable, preceded by a caret (^).

```
EType=start, AType=exec, Action="/tmp/trigger2.sh", Requires=^Var1
```

*The trigger searches through the parent objects in which it resides for the variable Var1.*

## Requesting name space variables

### To request a name space variable in a trigger

- [Configure the trigger](#). If it is attached to a generic system job, verify that it meets all the generic system job trigger requirements.
- Create an argument list in the **Action** attribute (after the script path and before the closing quotes) and request the desired variable with an asterisk (\*) in place of the name space.

```
...Action="$HOME/myTrig.py $*.IPAddr"...
```

*Each applicable name space variable is added to the argument list in the format <varName>=<val>.*

For instance, the example above would cause the script to run the following way:

```
> myTrig.py vc1.IPAddr=/tmp/dir1 vc2.IPAddr=/tmp/dir2 vc4.IPAddr=/tmp/dir3
```

*Any other arguments provided here without name spaces will not change.*

- Filter which name spaces are passed down to a job trigger by setting **trigns** when you submit the job. Its value is a comma-delimited list of the desired name spaces.

```
msub -l ... -W x="trigns=vc2,vc4"
```

If the new job is applied to the example in step 2, the script's arguments include *vc2.IPAddr* and *vc4.Addr* and exclude *vc1.IPAddr*. The script runs as follows:

```
> myTrig.py vc2.IPAddr=/tmp/dir1 vc4.IPAddr=/tmp/dir2
```

## References

### Dependency trigger components

Sets	
<b>Possible values</b>	' ' delimited string
<b>Description</b>	Variable values this trigger sets upon success or failure
<b>Usage notes</b>	Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the trigger completes and satisfies all its set conditions. Used in conjunction with <a href="#">Requires on page 779</a> to create trigger dependencies.

Unsets	
<b>Possible values</b>	' ' delimited string
<b>Description</b>	Variable this trigger destroys upon success or failure.
<b>Usage notes</b>	Preceding the string with an exclamation mark (!) indicates this variable is unset upon trigger failure. Used in conjunction with <a href="#">Requires on page 779</a> to create trigger dependencies.

Requires	
<b>Possible values</b>	' ' delimited string
<b>Description</b>	Variables this trigger requires to be set or not set before it will fire.
<b>Usage notes</b>	Preceding the string with an exclamation mark (!) indicates this variable must not be set. Preceding the string with a caret (^) indicates that the variable may come from a parent object (See <a href="#">Requiring variables from generations of parent objects on page 778</a> for more information.). Used in conjunction with <a href="#">Sets on page 779</a> to create trigger dependencies.

### Trigger variable comparison types

The following table describes the valid types of comparisons you can use to express the relationship of a trigger variable to its value:

Type	Comparison	Notes
<b>set</b>	is set (exists)	Default
<b>notset</b>	not set (does not exist)	Same as specifying '!' before a variable
<b>eq</b>	equals	
<b>ne</b>	not equal	
<b>gt</b>	greater than	Integer values only
<b>lt</b>	less than	Integer values only
<b>ge</b>	greater than or equal to	Integer values only
<b>le</b>	less than or equal to	Integer values only

### Internal variables

Several internal variables are available for use in trigger scripts. These can be accessed using `$(varName)`.

Internal Variables	
<b>ETYPE</b>	The type of event that signals that the trigger can fire. ETYPE values include cancel, checkpoint, create, end, fail, hold, migrate, preempt, standing, start, and threshold.
<b>OID</b>	The name of the object to which the trigger was attached
<b>OTYPE</b>	The type of object to which the trigger is attached; can be rsv, job, node, vm, or sched
<b>OWNERMAIL</b>	A variable that is populated only if the trigger's parent object has a user associated with it and that user has an email address associated with it
<b>TIME</b>	The time of the trigger launch in the following format: Wed Mar 10 12:35:12 2012
<b>USER</b>	The user (when applicable)

## Object-specific internal variables

Job Variables	
<b>MASTERHOST</b>	The primary node for the job
<b>HOSTLIST</b>	The entire hostlist of the job

Reservation Variables	
<b>HOSTLIST</b>	The entire hostlist for the reservation
<b>OBJECTXML</b>	The XML representation of an object output is the same that is generated by <a href="#">mdiag -r --xml</a>
<b>OS</b>	The operating system on the first node of the reservation
<b>OWNER</b>	The owner of the reservation

Example 3-151: Internal variable example

```
AType=exec,Action="/tmp/trigger.sh $OID $HOSTLIST",EType=start
```

*The object ID (`$OID`) and hostlist (`$HOSTLIST`) will be passed to `/tmp/trigger.sh` as command line arguments when the trigger executes the script. The script can then process this information as needed.*

## Miscellaneous

- [User Feedback Overview](#) on page 781
- [Enabling High Availability Features](#) on page 783
- [Malleable Jobs](#) on page 785
- [Identity Managers](#) on page 786
- [Generic System Jobs](#) on page 790

## User Feedback Overview

The Feedback facility allows a site administrator to provide job performance information to users at job completion time. When a job completes, the program pointed to by the [FEEDBACKPROGRAM](#) parameter is called with a number of command line arguments. The site administrator is responsible for creating a program capable of processing and acting upon the contents of the command line. The command line arguments passed are as follows:

1. job id
2. user name
3. user email
4. final job state
5. QoS requested
6. epoch time job was submitted
7. epoch time job started
8. epoch time job completed
9. job XFactor
10. job wallclock limit
11. processors requested
12. memory requested
13. average per task cpu load
14. maximum per task cpu load
15. average per task memory usage
16. maximum per task memory usage
17. messages associated with the job
18. hostlist (comma-delimited)

For many sites, the feedback script is useful as a means of letting users know the accuracy of their wallclock limit estimate, as well as the CPU efficiency, and memory usage pattern of their job. The feedback script may be used as a mechanism to do any of the following:

- email users regarding statistics of all completed jobs
- email users only when certain criteria are met (such as "Job 14991 has just completed which requested 128 MB of memory per task. During execution, it used 253 MB of memory per task potentially conflicting with other jobs. Please improve your resource usage estimates in future jobs.")
- update system databases
- take system actions based on job completion statistics



Some of these fields may be set to zero if the underlying OS/resource manager does not support the necessary data collection.

*Example 3-152:*

```
FEEDBACKPROGAM /opt/moab/tools/fb.pl
```

## Enabling High Availability Features

**i** Contact Adaptive Computing before attempting to implement any type of high availability.

- [Moab High Availability Overview](#)
  - [Configuring High Availability via a Networked File System](#)
  - [Confirming High Availability on a Networked File System](#)
- [Other High Availability Configuration](#)

### High Availability Overview

High availability allows Moab to run on two different machines: a primary and secondary server. The configuration method to achieve this behavior takes advantage of a networked file system to configure two Moab servers with only one operating at a time.

**i** If you use a shared file system for high availability and Moab is configured to use a database, Moab must be an ODBC build, not SQLite.

When configured to run on a networked file system — any networked file system that supports file locking is supported — the first Moab server that starts locks a particular file. The second Moab server waits on that lock and only begins scheduling when it gains control of the lock on the file. This method achieves near instantaneous turnover between failures and eliminates the need for two Moab servers to synchronize information periodically as the two Moab servers access the same database/checkpoint file.

**i** As Moab uses timestamping in the lock file to implement high availability, the clocks on both servers require synchronization; all machines in a cluster must be synchronized to the same time server.

Moab high availability and TORQUE high availability operate independently of each other. If a job is submitted with `msub` and the primary Moab server is down, `msub` tries to connect to the fallback Moab server. Once the job is given to TORQUE, if TORQUE can't connect to the primary `pbs_server`, it tries to connect to the fallback `pbs_server`. For example:

A job is submitted with `msub`, but Moab is down on `server01`, so `msub` contacts Moab running on `server02`.

A job is submitted with `msub` and Moab hands it off to TORQUE, but `pbs_server` is down on `server01`, so `qsub` contacts `pbs_server` running on `server02`.

When you shut down or restart Moab on both servers, you must run the command twice. A single shutdown (`mschedctl -k`) or restart (`mschedctl -R`) command will go to the primary server and kill it, causing the secondary server to fall back and start operating. To kill the secondary server, resubmit the command.

**i** Do not use anything but a plain simple NFS fileshare that is not used by anybody or anything else (i.e., only Moab can use the fileshare).

**i** Do not use any general-purpose NAS, do not use any parallel file system, and do not use company-wide shared infrastructure to set up Moab high availability using "native" high availability.

## Configuring High Availability on a Networked File System

Because the two Moab servers access the same files, configuration is only required in the `moab.cfg` file. The two hosts that run Moab must be configured with the **SERVER** and **FBSERVER** parameters. File lock is turned on using the **FLAGS=filelockha** flag. Specify the lock file with the **HALOCKFILE** parameter. The following example illustrates a possible configuration:

```
SCHEDCFG [Moab]  SERVER=host1:42559
SCHEDCFG [Moab]  FBSERVER=host2
SCHEDCFG [Moab]  FLAGS=filelockha
SCHEDCFG [Moab]  HALOCKFILE=/opt/moab/.moab_lock
```

Use the **HALOCKUPDATETIME** parameter to specify how frequently the primary server updates the timestamp on the lock file. Use the **HALOCKCHECKTIME** parameter to specify how frequently the secondary server checks the timestamp on the lock file.

```
HALOCKCHECKTIME 9
HALOCKUPDATETIME 3
```

*In the preceding example, the secondary server checks the lock file for updates every 9 seconds. The **HALOCKUPDATETIME** parameter is set to 3 seconds, permitting the primary server three opportunities to update the timestamp for each time the secondary server checks the timestamp on the lock file.*

**i** **FBSERVER** does not take a port number. The primary server's port is used for both the primary server and the fallback server.

## Confirming High Availability on a Networked File System

Administrators can run the `mdiag -S -v` command to view which Moab server is currently scheduling and responding to client requests.

## Other High Availability Configuration

Moab has many features to improve the availability of a cluster beyond the ability to automatically relocate to another execution server. The following table describes some of these features.

Feature	Description
<b>AMCFG[] BACKUPHOST</b>	<p>If using the Moab Accounting Manager, you may enable high availability with the accounting manager by specifying a backup server as in the following example:</p> <pre>AMCFG[mam] BACKUPHOST=headnode2</pre>
<b><u>JOBACNONNODEFAILURE</u></b>	<p>If a node allocated to an active job fails, it is possible for the job to continue running indefinitely even though the output it produces is of no value. Setting this parameter allows the scheduler to automatically preempt these jobs when a node failure is detected, possibly allowing the job to run elsewhere and also allowing other allocated nodes to be used by other jobs.</p>
<b><u>SCHEDCFG[] RECOVERYACTION</u></b>	<p>If a catastrophic failure event occurs (SIGSEGV or SIGILL signal is triggered), Moab can be configured to automatically restart, trap the failure, ignore the failure, or behave in the default manner for the specified signal. These actions are specified using the values <i>RESTART</i>, <i>TRAP</i>, <i>IGNORE</i>, or <i>DIE</i>, as in the following example:</p> <pre>SCHEDCFG[bas] MODE=NORMAL RECOVERYACTION=RESTART</pre>

## Malleable Jobs

Malleable jobs are jobs that can be adjusted in terms of resources and duration required, and which allow the scheduler to maximize job responsiveness by selecting a job's resource shape or footprint prior to job execution. Once a job has started, however, its resource footprint is fixed until job completion.

To enable malleable jobs, the underlying resource manager must support dynamic modification of resource requirements prior to execution (i.e., [TORQUE](#)) and the jobs must be submitted using the [TRL](#) (task request list) resource manager extension string. With the **TRL** attribute specified, Moab will attempt to select a start time and resource footprint to minimize job completion time and maximize overall effective system utilization (i.e.,  $\langle \text{AverageJobEfficiency} \rangle * \langle \text{AverageSystemUtilization} \rangle$ ).

*Example 3-153:*

With the following job submission, Moab will execute the job in one of the following configurations: 1 node for 1 hour, 2 nodes for 30 minutes, or 4 nodes for 15 minutes.

```
> qsub -l nodes=1,trl=1@3600:2@1800:4@900 testjob.cmd
job 72436.orion submitted
```

## Identity Managers

- [Identity Manager Overview](#)
- [Basic Configuration](#)
- [Importing Credential Fairness Policies](#)
- [Identity Manager Data Format](#)
- [Identity Manager Conflicts](#)
- [Refreshing Identity Manager Data](#)

The Moab identity manager interface can be used to coordinate global and local information regarding users, groups, accounts, and classes associated with compute resources. The identity manager interface may also be used to allow Moab to automatically and dynamically create and modify user accounts and credential attributes according to current workload needs.

 Only one identity manager can be configured at a time.

### Identity Manager Overview

Moab allows sites extensive flexibility when it comes to defining credential access, attributes, and relationships. In most cases, use of the [USERCFG](#), [GROUPCFG](#), [ACCOUNTCFG](#), [CLASSCFG](#), and [QOSCFG](#) parameters is adequate to specify the needed configuration. However, in certain cases such as the following, this approach may not be ideal or even adequate:

- Environments with very large user sets
- Environments with very dynamic credential configurations in terms of fairshare targets, priorities, service access constraints, and credential relationships
- Grid environments with external credential mapping information services
- Enterprise environments with fairness policies based on multi-cluster usage

Moab addresses these and similar issues through the use of an identity manager. An identity manager is configured with the [IDCFG](#) parameter and allows Moab to exchange information with an external identity management service. As with Moab resource manager interfaces, this service can be a full commercial package designed for this purpose, or something far simpler such as a web service, text file, or database.

### Basic Configuration

Configuring an identity manager in basic read-only mode can be accomplished by simply setting the **SERVER** attribute. If Moab is to interact with the identity manager in read/write mode, some additional configuration may be required.

BLOCKCREDLIST	
<b>Format</b>	One or more comma-delimited object types from the following list: <i>acct</i> , <i>group</i> , or <i>user</i>
<b>Details</b>	<p>If specified, Moab will block all jobs associated with credentials not explicitly reported in the most recent identity manager update. If the credential appears on subsequent updates, resource access will be immediately restored.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin: 5px 0;"> <p><b>i</b> Jobs will only be blocked if <u>fairshare</u> is enabled. This can be accomplished by setting the <u>FSPOLICY</u> parameter to any value such as in the following example:</p> <pre style="border: 1px dashed #0070C0; padding: 5px; display: inline-block;">FSPOLICY DEDICATEDPS</pre> </div>
<b>Example</b>	<pre>IDCFG[test01] BLOCKCREDLIST=acct,user,groups</pre> <div style="border: 1px dashed #0070C0; padding: 5px; margin: 5px 0;"> <p><i>Moab will block any jobs associated with accounts, users, or groups not in the most recent identity manager update.</i></p> </div>

CREATECRED	
<b>Format</b>	<BOOLEAN> (default is <i>FALSE</i> )
<b>Details</b>	Specifies whether Moab should create credentials reported by the identity manager that have not yet been locally discovered or loaded via the resource manager. By default, Moab will only load information for credentials which have been discovered outside of the identity manager.
<b>Example</b>	<pre>IDCFG[test01] CREATECRED=TRUE</pre> <div style="border: 1px dashed #0070C0; padding: 5px; margin: 5px 0;"> <p><i>Moab will create credentials from test01 that have not been previously loaded.</i></p> </div>

REFRESHPERIOD	
<b>Format</b>	<i>minute</i> , <i>hour</i> , <i>day</i> , or <i>infinity</i> (default is <i>infinity</i> ).
<b>Details</b>	If specified, Moab refreshes identity manager information once every specified iteration. If <i>infinity</i> is specified, the information is updated only at Moab start up.
<b>Example</b>	<pre>IDCFG[test01] REFRESHPERIOD=hour</pre> <div style="border: 1px dashed #0070C0; padding: 5px; margin: 5px 0;"> <p><i>Moab queries the identity manager every hour.</i></p> </div>

RESETCREDLIST	
<b>Format</b>	One or more comma-delimited object types from the following list: <i>acct</i> , <i>group</i> , or <i>user</i> .
<b>Details</b>	If specified, Moab will reset the account access list and fairshare cap and target for all credentials of the specified type(s) regardless of whether they are included in the current info manager report. Moab will then load information for the specified credentials.
<b>Example</b>	<pre>IDCFG[test01] RESETCREDLIST=group</pre> <p><i>Moab will reset the account access list and fairshare target for all groups.</i></p>

SERVER	
<b>Format</b>	<URL>
<b>Details</b>	Specifies the protocol/interface to use to contact the identity manager.
<b>Example</b>	<pre>IDCFG[test01] SERVER=exec://\$HOME/example.pl</pre> <p><i>Moab will use example.pl to communicate with the identity manager.</i></p>

UPDATEREFRESHONFAILURE	
<b>Format</b>	<BOOLEAN> (default is <i>FALSE</i> )
<b>Details</b>	When an <b>IDCFG</b> script fails, it retries almost immediately and continuously until it succeeds. When <b>UPDATEREFRESHONFAILURE</b> is set to <i>TRUE</i> , a failed script does not attempt to rerun immediately, but instead follows the specified <b>REFRESHPERIOD</b> schedule. When set to <i>TRUE</i> , <b>UPDATEREFRESHONFAILURE</b> updates the script execution timestamp, even if the script does not end successfully.
<b>Example</b>	<pre>IDCFG[info] SERVER=exec:///home/tshaw/test/1447/bad_script.pl REFRESHPERIOD=hour UPDATEREFRESHONFAILURE=TRUE</pre>

## Importing Credential Fairness Policies

One common use for an identity manager is to import fairness data from a global external information service. As an example, assume a site needed to coordinate Moab group level fairshare targets with an

allocation database that constrains total allocations available to any given group. To enable this, a configuration like the following might be used:

```
IDCFG[alloc] SERVER=exec://$TOOLS_DIR/idquery.pl
...
```

*The tools/idquery.pl script could be set up to query a local database and report its results to Moab. Each iteration, Moab will then import this information, adjust its internal configuration, and immediately respect the new fairness policies.*

## Identity Manager Data Format

When an identity manager outputs credential information either through an `exec` or file based interface, the data should be organized in the following format:

```
<CREDTYPE>:<CREDID> <ATTR>=<VALUE>
```

where

- `<CREDTYPE>` is one of `user`, `group`, `account`, `class`, or `qos`.
- `<CREDID>` is the name of the credential.
- `<ATTR>` is one of `adminlevel`, `alist`, `chargerate`, [comment](#), [emailaddress](#), `fstarget`, `globalfstarget`, `globalfsusage`, [maxjob](#), [maxmem](#), [maxnode](#), [maxpe](#), [maxproc](#), [maxps](#), [maxwc](#), `plist`, `priority`, `qlist`, or `role`. [Multi-dimensional policies](#) work here as well.
- `<VALUE>` is the value for the specified attribute.



To clear a comment, set its value to `""`; for example: `comment=""`.

*Example 3-154:*

The following output may be generated by an `exec` based identity manager:

```
group:financial fstarget=16.3 alist=project2
group:marketing fstarget=2.5
group:engineering fstarget=36.7
group:dm fstarget=42.5
user:jason adminlevel=3
account:sales maxnode=128 maxjob=8,16
```

The following example limits user `bob` to 8 `matlab` generic resources.

```
user:bob MAXGRES[matlab]=8
```



To specify unlimited use of generic resources, set the value to `-1`.

## Identity Manager Conflicts

When local credential configuration (as specified via `moab.cfg`) conflicts with identity manager configuration, the identity manager value takes precedence and the local values are overwritten.

## Refreshing Identity Manager Data

By default, Moab only loads identity manager information once when it is first started up. If the identity manager data is dynamic, then you may want Moab to periodically update its information. To do this, set the **REFRESHPERIOD** attribute of the **IDCFG** parameter. Legal values are documented in the following table:

Value	Description
<b>minute</b>	Update identity information once per minute
<b>hour</b>	Update identity information once per hour
<b>day</b>	Update identity information once per day
<b>infinity</b>	Update identity information only at start-up (default)

Example 3-155:

```
IDCFG[hq] SERVER=exec://$TOOLS DIR/updatepolicy.sh REFRESHPERIOD=hour
```

 Job credential feasibility is evaluated at job submission and start time.

### Related topics

- [Credential Overview](#)
- [Usage Limits/Throttling Policies](#)

## Generic System Jobs

Generic system jobs are system jobs with a trigger. They are useful for specifying steps in a workflow.

- [Creating a Generic System Job](#)
  - [The Trigger](#)
- [Workflows Using Job Template Dependencies](#)
  - [Inheriting Resources in Workflows](#)

### Creating a Generic System Job

Generic system jobs are specified via a job template. The template can be selectable and you must use the **GENERICSYSJOB** attribute to let Moab know that this job template describes a generic system job and to specify a trigger, as shown in the following example:

```
JOBCFG[gen]
GENERICSYSJOB=EType=start,AType=exec,Action="$HOME/genericTrig.py",Timeout=5:00
```

## The Trigger

The generic system job's trigger that meets certain criteria. This trigger must have a timeout, an `Atype=Exec`, and the `EType` must equal "start". The timeout of the trigger will be used as the walltime for the job. The trigger will begin when the system job begins and the job will be considered completed when the trigger completes. The job will have the same completion code as the trigger. The walltime on the job template is not applicable in this case since the timeout of the trigger will be the walltime.

If the trigger fails, an error message will be attached to all of the job's parent VCs. You can view this in the `--xml` output of the VC query. The message includes the location of `STDIN`, `STDOUT`, and `STDERR` files. For example:

```

mvcctl -q ALL --xml
<Data>
<vc CREATETIME="1320184350" DESCRIPTION="Moab.1"
  FLAGS="DESTROYOBJECTS, DESTROYWHENEMPTY, HASSTARTED, WORKFLOW"
  JOBS="Moab.1" NAME="vc1" OWNER="user:frank">
<ACL aff="positive" cmp="%=" name="frank" type="USER"></ACL>
<MESSAGES>
<message COUNT="1" CTIME="1320184362"
  DATA="Trigger 10 failed on job Moab.1.setup- STDIN:
/tmp/ByLLl2wv/spool/vm.py.ieWPPS5 STDOUT:
/tmp/ByLLl2wv/spool/vm.py.oDMIXAW STDERR /tmp/ByLLl2wv/spool/vm.py.e2jD5iN"
  EXPIRETIME="1322776362" OWNER="frank" PRIORITY="0"
  TYPE="other" index="0"></message>
</MESSAGES>
<Variables>
<Variable name="VMID">vm1</Variable>
<Variable name="HV">TRUE</Variable>
</Variables>
</vc>
</Data>

```

You can specify other triggers on a generic system job using the **TRIGGER** attribute and delimiting them with semicolons. For example:

```

JOBCFG[gen]  GENERICSYSJOB=<genericSystemJobTriggerSpecs>
JOBCFG[gen]  TRIGGER=<triggerSpecs>;TRIGGER=<triggerSpecs>

```

## Workflows Using Job Template Dependencies

To create workflows, use the following format:

```

JOBCFG[gen]  TEMPLATEDEPEND=AFTERANY:otherTemplate

```

This will create a job based on the template `otherTemplate`. The generic job will run after the `otherTemplate` job has finished. [Afterany](#) in the example means after all other jobs have completed, regardless of success.

## Inheriting Resources in Workflows

The **INHERITRES** flag can be used to cause the same resources in one step of a workflow to be passed to the next step:

```

JOBcfg[gen]   TEMPLATEDEPEND=AFTERANY:otherTemplate
JOBcfg[otherTemplate] INHERITRES=TRUE

```

*This example forces the job based on `otherTemplate` to have the same resource requirements as its parent. When the `otherTemplate` job is finished, the **INHERITRES** flag will cause the parent to run on the same resources as the child.*

The job that finishes first will pass its allocation up.

Any variables on the original job will be passed to the other jobs in the workflow. Variables can be added by other jobs in the workflow via the [sets](#) attribute in the generic system job's trigger. Other triggers must then request that variable name in the command line options.

**i** You will need to set the carat (^) in order for the variable to be sent up to the job group.

If you set the variable, you need to set it in the STDOUT of the trigger script. See the example below:

```

JOBcfg[W1]   GENERICSYSJOB=...,action='$HOME/W1.py $ipaddress' TEMPLATEDEPEND=AFTER:W2
JOBcfg[W2]   TRIGGER=...,action='$HOME/W2.py',sets=^ipaddress

```

*If a variable value is not set in STDOUT, it will be set to **TRUE**.*

To set the variable to a specific value, the `w2.py` script must set the value in its STDOUT:

```

print "ipaddress=10.10.10.1" #This will be parsed by Moab and set as the value of the
"ipaddress" variable

```

#### Example 3-156:

To create a VM with a workflow using job template dependencies and generic system jobs, use the following format:

```

#The job template that is "gate" to the workflow
JOBcfg[CreateVMWithSoftware] TEMPLATEDEPEND=AFTEROK:InstallSoftware SELECT=TRUE

JOBcfg[InstallSoftware]
GENERICSYSJOB=EType=start,AType=exec,Action="$HOME/setupSoftware.py
$IPAddr",Timeout=30:00
JOBcfg[InstallSoftware]   INHERITRES=TRUE
JOBcfg[InstallSoftware]   TEMPLATEDEPEND=AFTEROK:CreateVM

JOBcfg[CreateVM]   GENERICSYSJOB=EType=start,AType=exec,Action=$HOME/installVM.py
$HOSTLIST",Timeout=1:00:00,sets=^IPAddr
JOBcfg[CreateVM]   INHERITRES=TRUE

```

The user will then submit the job requesting what they need in the VM:

```

msub -l walltime=2:00:00,template=CreateVMWithSoftware,nodes=1:ppn=4,mem=1024
ActualWorkload.py

```

*The job will have the `CreateVMWithSoftware` template applied to it and will create the `InstallSoftware` job. The `InstallSoftware` job, because of **INHERITRES**, will have the same resource request (4 procs, 1GB of memory). This job then has its template applied to it which will do the same thing in creating the `CreateVM` job. The `CreateVM` job will then run, the trigger script will return the IP address of the new VM and pass its allocation up to the `InstallSoftware` job. The `InstallSoftware` job will use the `IPAddr` variable to find the VM and install the software. It will then return its resources up to the parent job, which will run the actual workload.*

## Database Configuration

Moab supports connecting to a database via native SQLite3, and it can also connect to other databases using the ODBC driver. These optional external databases store some additional information that the MongoDB database does not and allow you to query them directly using SQL. These databases are slower, however, and only SQLite3, which does not allow external queries, is supported.

The SQLite3 connection is for storing statistics. Consider reviewing the SQLite web page [Appropriate Uses for SQLite](#) for information regarding the suitability of using SQLite3 on your system.

While the ODBC connection is useful for storing statistics, it also stores events, nodes, and jobs. You can further configure Moab to store checkpoint information to a database rather than to the flat text file (`.moab.ck`) if you set the `CHECKPOINTWITHDATABASE` parameter to `TRUE`.

Connecting to an external database makes Moab more searchable, allowing you to run queries for statistics and events rather than using regular expressions to draw the information from the Moab flat files.

- [SQLite3 on page 793](#)
- [Connecting to a MySQL Database with an ODBC Driver on page 794](#)
- [Connecting to a PostgreSQL Database with an ODBC Driver on page 797](#)
- [Connecting to an Oracle Database with an ODBC Driver on page 799](#)
  - [Installing the Oracle Instant Client on page 806](#)
- [Migrating Your Database to Newer Versions of Moab on page 808](#)
- [Importing Statistics from stats/DAY.\\* to the Moab Database on page 815](#)

## SQLite3

Moab supports connecting to a database via native SQLite3. Database installation and configuration occurs automatically during normal Moab installation (`configure`, `make install`). If you did not follow the normal process to install Moab and need to install the database, do the following to manually install and configure Moab database support:

1. Create the database file `moab.db` in your moab home directory by running the following command from the root of your unzipped Moab build directory:
 

```
perl buildutils/install.sqlite3.pl <moab-home-directory>
```

  - Verify that the command worked by running `lib/sqlite3 <moab-home-directory>/moab.db`; at the resulting prompt, type `.tables` and press **ENTER**. You should see several tables such as `mcheckpoint` listed. Exit from this program with the `.quit` command.
  - The `perl buildutils/install.sqlite3.pl <moab-home-directory>` command may fail if your operating system cannot find the SQLite3 libraries. Also, Moab fails if unable to identify the libraries. To temporarily force the libraries to be found, run the following command:
 

```
export LD_LIBRARY_PATH=<location where libraries were copied>
```

2. In the `moab.cfg` file in the `etc/` folder of the home directory, add the following line:

```
USEDATABASE INTERNAL
```

To verify that Moab is running with SQLite3 support, start Moab and run the `mdiag -S -v` command. If there are no database-related error messages displayed, then Moab should be successfully connected to a database.

**i** > `moabd` is a safe and recommended method of starting Moab if things are not installed in their default locations.

## Connecting to a MySQL Database with an ODBC Driver

This documentation shows how to set up and configure Moab to connect to a MySQL database using the MySQL ODBC driver. This document assumes the necessary MySQL and ODBC drivers have already been installed and configured.

To set up and configure Moab to connect to a MySQL database using the MySQL ODBC driver, do the following:

**i** This solution has been tested and works with these versions:

- libmyodbc - 5.1.5
- MySQL 5.1

1. Download and install the ODBC version of Moab. [Install and configure](#) Moab as normal but add the following in the Moab configuration file (`moab.cfg`):

```
USEDATABASE          ODBC
# Turn on stat profiling
USERCFG [DEFAULT]    ENABLEPROFILING=TRUE
GROUPCFG [DEFAULT]   ENABLEPROFILING=TRUE
QOSCFG [DEFAULT]     ENABLEPROFILING=TRUE
CLASSCFG [DEFAULT]   ENABLEPROFILING=TRUE
ACCOUNTCFG [DEFAULT] ENABLEPROFILING=TRUE
NODECFG [DEFAULT]    ENABLEPROFILING=TRUE
```

2. Create the database in MySQL using the MySQL database dump contained in the `moab-db.sql` file. This file is located in the `contrib/sql` directory in the root of the binaries.

- Run the following command:

```
mysql -u root -p < moab-db-mysql-create.sql
```

3. Configure the MySQL and ODBC driver. The `odbcinst.ini` file must be contained in `/etc`.

**i** Run the following command to find the MySQL ODBC client driver. You could also query the `libmyodbc` package that was installed.

```
[root]# updatedb
[root]# locate libmyodbc
```

```
[MySQL]
Description = ODBC for MySQL
Driver = /usr/lib/odbc/libmyodbc.so
```

4. Configure Moab to use the MySQL ODBC driver. Moab uses an ODBC datastore file to connect to MySQL using ODBC. This file must be located in the Moab home directory (`/opt/moab` by default) and be named `dsninfo.dsn`, which is used by Moab. You need to have the following data in both `/etc/odbc.ini` and `$MOABHOMEDIR/dsninfo.dsn`:

```
[ODBC]
Driver = MySQL
USER = <username>
PASSWORD = <password>
Server = localhost
Database = Moab
Port = 3306
```

**i** The user should have read/write privileges on the Moab database.

The preceding example file tells ODBC to use the MySQL driver, username `<username>`, password `<password>`, and to connect to MySQL running on the localhost on port 3306. ODBC uses this information to connect to the database called `Moab`.

5. Test the ODBC to MySQL connection by running the `isql` command, which reads the `/etc/odbc.ini` file:

```

$ isql -v ODBC
+-----+
| Connected! |
| |
| sql-statement |
| help [tablename] |
| quit |
| |
+-----+
SQL> show tables;
+-----+
| Tables_in_Moab |
+-----+
| EventType |
| Events |
| GeneralStats |
| GenericMetrics |
| Moab |
| NodeStats |
| NodeStatsGenericResources |
| ObjectType |
| mcheckpoint |
+-----+
SQLRowCount returns 10
10 rows fetched
SQL>

```

If you encounter any errors using the `isql` command, there was a problem setting up the ODBC to MySQL connection. Try the following debugging steps to resolve the issue:

- a. The `odbcinst.ini` and `odbc.ini` files are usually assumed to be located in `/etc`, but that is not always true. Use the `odbcinst -j` command to determine the assumed location of the files in your configuration.

```

[root#] odbcinst -j
unixODBC 2.2.12
DRIVERS.....: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc.ini
USER DATA SOURCES...: /home/adaptive/.odbc.ini

```

- b. Because `odbcinst.ini` and `odbc.ini` are expected in `/etc/unixODBC`, not `/etc`, move them from `/etc` to `/etc/unixODBC`.
- c. Use the `strace` command to determine where `isql` expects the `odbc.ini` and `odbcinst.ini` files. Note the location in which `isql` expects these files.

```

$ strace isql -v ODBC

```

6. With the ODBC driver configured, the database created, and Moab configured to use the database, start Moab for it to begin storing information in the created database.

 > `moabd` is a safe and recommended method of starting Moab if things are not installed in their default locations.

## Related topics

- [Importing Statistics to the Moab Database](#)

## Connecting to a PostgreSQL Database with an ODBC Driver

This documentation shows how to set up and configure Moab to connect to a PostgreSQL database using the ODBC driver. This document assumes the necessary ODBC drivers have already been installed and configured.

**i** Occasionally vacuuming your PostgreSQL database could improve Moab performance. See the PostgreSQL documentation for information on how to vacuum your database.

To set up and configure Moab to connect to a PostgreSQL database using the ODBC driver, do the following:

**i** This solution has been tested and works with the following file version:

- odbc-postgresql - 1:08.03.0200-1.2

1. Configure the PostgreSQL and ODBC driver. `odbcinst.ini` file must be contained in `/etc`.

**i** Run the following commands to find the PostgreSQL ODBC client driver and setup file. You could also query the `libodbcpsql` package that was installed.

```
[root]# updatedb
[root]# locate psqlodbc
[root]# locate libodbcpsql
```

```
[PostgreSQL]
Description = PostgreSQL ODBC driver
Driver = /usr/lib/odbc/psqlodbc.so
Setup = /usr/lib/odbc/libodbcpsqlS.so
Debug = 0
CommLog = 1
UsageCount = 2
```

2. Configure Moab to use the PostgreSQL ODBC driver. Moab uses an ODBC data source file to connect to PostgreSQL using ODBC. This file must be located in the Moab home directory (`/opt/moab` by default) and be named `dsninfo.dsn`, which is used by Moab. If the following content, which follows the standard ODBC driver file syntax, is not already included in the `/etc/odbc.ini` file, make sure that you include it. Also, include the same content in the `dsninfo.dsn` file.

```
[ODBC]
Driver = PostgreSQL
Description = PostgreSQL Data Source
Servername = localhost
Port = 5432
Protocol = 8.4
Username = postgres
Password = moab
Database = Moab
```

**i** The user should have read/write privileges on the Moab database.

The preceding example file tells ODBC to use the PostgreSQL driver, `postgres` user, `moab` password, and to connect to PostgreSQL running on the localhost on port 5432. ODBC uses this information and connects to the database called `Moab`.

3. Test the ODBC to PostgreSQL connection by running the `isql` command, which reads the `/etc/odbc.ini` file. If connected, you should be able to run the `help` command.

If you encounter any errors using the `isql` command, there was a problem setting up the ODBC to MySQL connection. Try the following debugging steps to resolve the issue:

- a. The `odbcinst.ini` and `odbc.ini` files are usually assumed to be located in `/etc`, but that is not always true. Use the `odbcinst -j` command to determine the assumed location of the files in your configuration.

```
[root#] odbcinst -j
unixODBC 2.2.12
DRIVERS.....: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc.ini
USER DATA SOURCES...: /home/adaptive/.odbc.ini
```

- b. Because `odbcinst.ini` and `odbc.ini` are expected in `/etc/unixODBC`, not `/etc`, move them from `/etc` to `/etc/unixODBC`.
- c. Use the `strace` command to determine where `isql` expects the `odbc.ini` and `odbcinst.ini` files. Note the location in which `isql` expects these files.

```
$ strace isql -v ODBC
```

4. Create the database in PostgreSQL using the `moab-db-postgresql.sh` setup script contained in the `contrib/sql` directory at the root of the binary.
  - Run the script and provide the DB username that will attach to the Moab database (you must supply a DB username or the script will exit). The default admin user is `postgres`, but you can make a new user at this time:

```
> ./moab-db-postgresql.sh postgres
Create db user "postgres" in postgresQL? (y/n)>
```

- The script asks if you want to create the DB user you specified in postgresQL. If the DB user already exists, answer 'n'. Otherwise, the DB user is created and it asks for the new user's password.
  - The script then creates the database "Moab".
  - Finally, as the DB user you provided, the script imports the DB schema from `moab-db-postgresql-create.sql` into the Moab database.
5. Download and install the ODBC version of Moab. [Install and configure](#) Moab as normal but add the following in the Moab configuration file (`moab.cfg`):

```

USEDATABASE          ODBC
# Turn on stat profiling
USERCFG [DEFAULT]    ENABLEPROFILING=TRUE
GROUPCFG [DEFAULT]   ENABLEPROFILING=TRUE
QOSCFG [DEFAULT]     ENABLEPROFILING=TRUE
CLASSCFG [DEFAULT]   ENABLEPROFILING=TRUE
ACCOUNTCFG [DEFAULT] ENABLEPROFILING=TRUE
NODECFG [DEFAULT]    ENABLEPROFILING=TRUE

```

6. With the ODBC driver configured, the database created, and Moab configured to use the database, start Moab for it to begin storing information in the created database.

**i** > moabd is a safe and recommended method of starting Moab if things are not installed in their default locations.

#### Related topics

- [Importing Statistics to the Moab Database](#)

## Connecting to an Oracle Database with an ODBC Driver

### Context

This documentation shows how to set up and configure Moab to connect to an Oracle database using the ODBC driver.

### To connect to an Oracle database with an ODBC driver

1. Install and configure the Oracle Instant Client with ODBC supporting libraries. For instructions, see [Installing the Oracle Instant Client on page 806](#).
2. Open your Moab configuration file (`$MOABHOMEDIR/moab.cfg`) and add the following lines to the end of the file.

```

USEDATABASE ODBC
# Turn on stat profiling
USERCFG [DEFAULT]    ENABLEPROFILING=TRUE
GROUPCFG [DEFAULT]   ENABLEPROFILING=TRUE
QOSCFG [DEFAULT]     ENABLEPROFILING=TRUE
CLASSCFG [DEFAULT]   ENABLEPROFILING=TRUE
ACCOUNTCFG [DEFAULT] ENABLEPROFILING=TRUE
NODECFG [DEFAULT]    ENABLEPROFILING=TRUE

```

3. Configure the Oracle ODBC Driver. The `odbcinst.ini` file must be contained in `/etc`.

```
[root]# vim /etc/odbcinst.ini
```

**i** Run the following command to find the Oracle Instant Client driver. You could also query the Oracle Instant Client package that was installed.

```
[root]# updatedb && locate libsqora
```

Add the following text to the file.

```
[Oracle 11g ODBC driver]
Description      = Oracle ODBC driver for Oracle 11g
Driver          = /usr/lib/oracle/11.2/client64/lib/libsqora.so.11.1
Setup           =
FileUsage       =
CPTimeout       =
CPReuse         =
Driver Logging  = 7

[ODBC]
Trace = Yes
TraceFile = /tmp/odbc.log
ForceTrace = Yes
Pooling = No
DEBUG = 1
```

**i** Driver Logging is set high (level 7) so that you can debug during the installation and configuration process if necessary. You can decrease the setting or remove the directive once you finish the process.

**i** To configure the location of the ODBC log (/tmp/odbc.log), set the TraceFile attribute shown in the example above. See "[unixODBC without the GUI](#)" on the unixODBC website for more information.

4. Because the driver installed in step 1 is a shared library, run `ldd` to verify that it and all of its dependencies are installed and working.

```
[root]# ldd /usr/lib/oracle/11.2/client64/lib/libsqora.so.11.1
linux-vdso.so.1 => (0x00007fff631fff00)
libdl.so.2 => /lib64/libdl.so.2 (0x00007f8afbe83000)
libm.so.6 => /lib64/libm.so.6 (0x00007f8afbbff000)
libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f8afb9e1000)
libnsl.so.1 => /lib64/libnsl.so.1 (0x00007f8afb7c8000)
libclntsh.so.11.1 =>
/usr/lib/oracle/11.2/client64/lib/libclntsh.so.11.1 (0x00007f8af8e59000)
libodbcinst.so.1 => not found
libc.so.6 => /lib64/libc.so.6 (0x00007f8af8ac5000)
/lib64/ld-linux-x86-64.so.2 (0x0000003bdb000000)
libnnz11.so => /usr/lib/oracle/11.2/client64/lib/libnnz11.so
(0x00007f8af86f8000)
libaio.so.1 => /lib64/libaio.so.1 (0x00007f8af84f6000)
```

**i** If the command returns `libodbcinst.so.1 => not found`, create a symbolic link from `/usr/lib64/libodbcinst.so.1` to `/usr/lib64/libodbcinst.so.2`. This is a known Red Hat issue. See [Red Hat Bugzilla](#) for more information.

```
[root]# locate libodbcinst
/usr/local/lib/libodbcinst.so.2

[root]# cd /usr/lib64
[root]# ln -s libodbcinst.so.2 libodbcinst.so.1
```

Rerun `ldd`. It should load `libsqora.so.11.1` without error, as shown in the `ldd` example above.

**i** If the `ldd` command returns a warning like this: "ldd: warning: you do not have execution permission for `/usr/lib/oracle/11.2/client64/lib/libsqora.so.11.1`", run the following command:

```
[root]# chmod 755 /usr/lib/oracle/11.2/client64/lib/lib*
```

Rerun `ldd`. It should load `libsqora.so.11.1` without error, as shown in the `ldd` example above.

5. Configure Moab to use the Oracle ODBC driver. This example assumes that a Moab user exists and has been granted read and write privileges to the MOAB database instance referred to on the [Installing the Oracle Instant Client](#) on page 806 page.

```
[root]# vim $MOABHOMEDIR/dsninfo.dsn
```

Add the following lines the file, but change `ServerName`, `UserName`, and `Password` to suit your own system. `ServerName` is the name of the Oracle database instance. `Username` and `Password` are the credentials used to connect to that instance.

```
[ODBC]
Application Attributes = T
Attributes = W
BatchAutocommitMode = IfAllSuccessful
BindAsFLOAT = F
CloseCursor = F
DisableDPM = F
DisableMTS = T
Driver = Oracle 11g ODBC driver
DSN = ODBC
EXECSchemaOpt =
EXECSyntax = T
Failover = T
FailoverDelay = 10
FailoverRetryCount = 10
FetchBufferSize = 64000
ForceWCHAR = F
Lobs = T
Longs = T
MaxLargeData = 0
MetadataIdDefault = F
QueryTimeout = T
ResultSets = T
ServerName = MOAB
SQLGetData extensions = F
Translation DLL =
Translation Option = 0
DisableRULEHint = T
UserID = moab
Password = moab
StatementCache=F
CacheBufferSize=20
UseOCIDescribeAny=F
MaxTokenSize=8192
```

6. Add the contents of the `dsninfo.dsn` file to `/etc/odbc.ini`. Because the contents of `dsninfo.dsn` are required in both files, use the following command to concatenate the contents of `dsninfo.dsn` to `/etc/odbc.ini`. If the `odbc.ini` file already has content, verify that there are no conflicts.

```
[root]# cat $MOABHOMDIR/dsninfo.dsn >> /etc/odbc.ini
```

7. Create a directory to store the `tnsnames.ora` file you will create in the next step.

```
[root]# mkdir /etc/oracle
```

8. Create the `tnsnames.ora` file. The `ServerName` in `$MOABHOMEDIR/dsninfo.dsn` tells the Oracle ODBC driver what `tnsnames.ora` entry to use (`MOAB`). The `MOAB` `tnsnames` entry tells the Oracle ODBC driver to connect to server `adaptive-oracle` on the local domain (`ac`) on port 1561 using TCP and to connect to the Oracle instance named `MOAB` (The `SID` is the unique name of the instance).

```
[root]# cat >/etc/oracle/tnsnames.ora <<EOL
MOAB =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP) (HOST = adaptive-oracle) (PORT = 1561))
    )
    (CONNECT_DATA =
      (SID = MOAB)
    )
  )
EOL
```

9. Create a profile script (`oracle-instant-client.sh`) to be invoked by the operating system at startup. This script will set the `ORACLE_HOME`, `TWO_TASK`, and `TNS_ADMIN` environment variables required by Oracle and will amend the `LD_LIBRARY_PATH` to include required Oracle client libraries in the library search path.

```
[root]# cat >/etc/profile.d/oracle-instant-client.sh <<EOL
# Set ORACLE_HOME to the directory where the bin and lib directories are located
for the oracle client
export ORACLE_HOME=/usr/lib/oracle/11.2/client64

# No need to add ORACLE_HOME to the linker search path. oracle-instant-client.conf
in
# /etc/ld.so.conf.d should already contain /usr/lib/oracle/11.2/client64.
# Alternately, you can set it here by uncommenting the following line:
# export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ORACLE_HOME/lib

# Define the default location where Oracle should look for the server
export TWO_TASK=//adaptive-oracle:1561/listener

# Define where to find the tnsnames.ora file
export TNS_ADMIN=/etc/oracle
EOL
```

10. Source the `oracle-instant-client.sh` script and verify that each environment variable is set correctly.

```
[root]# source /etc/profile.d/oracle-instant-client.sh
[root]# echo $ORACLE_HOME
[root]# echo $LD_LIBRARY_PATH
[root]# echo $TWO_TASK
[root]# echo $TNS_ADMIN
```

11. Modify either the Moab startup script (`/etc/init.d/moab`) – recommended – or the `moabd` script (`/opt/moab/sbin/moabd`) to source `oracle-instant-client.sh`.

- Moab startup script (recommended): the following example suggests a location to source the `oracle-instant-client.sh` script within the Moab startup script.

```
...
# Export all environment variables required by the Oracle Instant Client
. /etc/profile.d/oracle-instant-client.sh

export MOABHOMEDIR=/opt/moab
...
```

- **moabd shell script:** the following example will resemble the moabd script in /opt/moab/sbin. Note that the moabd script is not invoked by the Moab startup script; The Moab startup script invokes the Moab binary (/opt/moab/sbin/moab) by default.

```
#!/bin/sh
#
# Copyright (C) 2012 by Adaptive Computing Enterprises, Inc. All Rights
Reserved.
#
# Export all environment variables required by the Oracle Instant Client
. /etc/profile.d/oracle-instant-client.sh
MOABHOMEDIR="/opt/moab" LD_LIBRARY_PATH="/opt/moab/lib:$LD_LIBRARY_PATH" moab
"$@"
```

## 12. Verify the Oracle ODBC driver is working.

```
isql -v ODBC
+-----+
| Connected!
|
| sql-statement
| help [tablename]
| quit
|
+-----+
```

If you encounter any errors using the `isql` command, there was a problem setting up the ODBC to Oracle connection. Try the following debugging steps to resolve the issue:

- The `odbcinst.ini` and `odbc.ini` files are usually assumed to be located in `/etc`, but that is not always true. Use the `odbcinst -j` command to determine the assumed location of the files in your configuration.

```
[root#] odbcinst -j
unixODBC 2.2.12
DRIVERS.....: /etc/unixODBC/odbcinst.ini
SYSTEM DATA SOURCES: /etc/unixODBC/odbc.ini
USER DATA SOURCES..: /home/adaptive/.odbc.ini
```

- Because `odbcinst.ini` and `odbc.ini` are expected in `/etc/unixODBC`, not `/etc`, move them from `/etc` to `/etc/unixODBC`.
- Use the `strace` command to determine where `isql` expects the `odbc.ini` and `odbcinst.ini` files. Note the location in which `isql` expects these files.

```
$ strace isql -v ODBC
```

- If you have not already done so, create the database tables in Oracle using the `moab-db-oracle-create.sql` script located in the `contrib/sql` directory in the root of the binaries. This example assumes that you are logged into the MOAB database instance (referred to on the [Installing the Oracle Instant Client on page 806](#) page) as Moab user with read and write privileges.

```
SQL> @./contrib/sql/moab-db-oracle-create.sql
```

- Verify that the database schema installed correctly by listing the tables. Your results should look like this:

```
SQL> select table_name from all_tables where owner = 'MOAB';
+-----+
| TABLE_NAME |
+-----+
| TRIGGERS     |
| MOAB        |
| OBJECTTYPE  |
| VCS         |
| EVENTTYPE   |
| JOBHISTORY  |
| MCHECKPOINT |
| NODES       |
| EVENTS      |
| NODESTATSGENERICRESOURCES |
| JOBS        |
| RESERVATIONS |
| GENERICMETRICS |
| REQUESTS    |
| GENERALSTATS |
| NODESTATS   |
+-----+
SQLRowCount returns -1
16 rows fetched
```

#### 15. Restart Moab.

```
[root]# mschedctl -R
```

#### 16. Verify Moab is correctly configured to write to the Oracle database by doing each of the following steps:

##### a. Tail the moab.log file for ODBC errors.

```
# Check the $MOABHOMEDIR/log/moab.log file for ODBC errors. You should see a few hits even if there are no errors.
[root]# tail -f $MOABHOMEDIR/log/moab.log | grep -i odbc
```

##### b. Log in to the Moab Oracle database.

In the first example below, `isql` will search `/etc/odbc.ini` for "[ODBC]". `unixODBC` will then use the Oracle 11g ODBC driver defined in `/etc/odbcinst.ini` to establish a connection. The `ServerName` in `/etc/odbc.ini` tells the Oracle driver to reference the `MOAB` `tnsnames` entry in `/etc/oracle/tnsnames.ora` for connection parameters.

The second example uses `sqlplus` and a connect string to connect.

Try both connection methods.

```
# Log in to Oracle. Try both isql and sqlplus64 clients.
[root]# isql -v ODBC
```

```
[root]# sqlplus64 moab/moab@adaptive-oracle:1561/MOAB
```

##### c. Select some data from one or more of the tables (Nodes, Events, and the like) to verify that data is being stored in the Moab Oracle instance.

```

# sqlplus64 moab/moab@adaptive-oracle:1561/MOAB

SQL*Plus: Release 11.2.0.4.0 Production on Fri Oct 4 14:59:02 2013

Copyright (c) 1982, 2013, Oracle. All rights reserved.

Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production

SQL> select table_name from user_tables;

TABLE_NAME
-----
JOBS
REQUESTS
RESERVATIONS
VCS
EVENTTYPE
GENERALSTATS
GENERICMETRICS
NODESTATS
NODESTATSGENERICRESOURCES
EVENTS
JOBHISTORY
MCHECKPOINT
NODES
TRIGGERS
MOAB
OBJECTTYPE

16 rows selected.

```

## Related topics

- [Installing the Oracle Instant Client on page 806](#)
- [Connecting to a MySQL Database with an ODBC Driver on page 794](#)
- [Connecting to a PostgreSQL Database with an ODBC Driver on page 797](#)
- [Database Configuration on page 793](#)

## Installing the Oracle Instant Client

### Context

The following procedure demonstrates how to install the correct ODBC drivers for your Oracle database. This guide is a prerequisite for the [Connecting to an Oracle Database with an ODBC Driver on page 799](#) task. Each step must be performed as root.

### To install the Oracle Instant Client

1. Go to the "[Install Client Downloads](#)" page on the Oracle website. Choose the link that matches your system type (for instance, Instant Client for Linux x86-64). Choose **Accept License Agreement** at the top of the page and download the following RPM or zip files for your target version (such as 11.2):

 The process of connecting Oracle to Moab Workload Manager has been tested on Oracle Instant Client version 11.2. The process may work with other versions, but they are not supported.

- Basic (oracle-instantclient11.2-basic-11.2.0.4.0-1.x86\_64.rpm)
- SQL Plus (oracle-instantclient11.2-sqlplus-11.2.0.4.0-1.x86\_64.rpm)
- ODBC (oracle-instantclient11.2-odbc-11.2.0.4.0-1.x86\_64.rpm)

2. Install the packages. This example installs the RPMs.

```
[root]# rpm -i ./oracle-instantclient11.2-basic-11.2.0.4.0-1.x86_64.rpm
[root]# rpm -i ./oracle-instantclient11.2-sqlplus-11.2.0.4.0-1.x86_64.rpm
[root]# rpm -i ./oracle-instantclient11.2-odbc-11.2.0.4.0-1.x86_64.rpm
```

3. Create a configuration file in /etc/ld.so.conf.d to add the Oracle client libraries to the LD\_LIBRARY\_PATH.

To confirm where the RPMs installed the libraries, run `rpm -qlp <rpmFileName>`.

```
[root]# cat >/etc/ld.so.conf.d/oracle-instant-client.conf <<EOL
/usr/lib/oracle/11.2/client64/lib
EOL
```

**i** If you installed Oracle Instant Client from a repository, run `repoquery -ql <rpmName>` instead.

Rebuild the LD\_LIBRARY\_PATH.

```
[root]# ldconfig
```

4. Connect to the database using `sqlplus`. If you used RPMs to install the client, the 32-bit and 64-bit clients are already in your PATH.

```
[root]# sqlplus64 moab/moab@adaptive-oracle:1561/MOAB
SQL*Plus: Release 11.2.0.4.0 Production on Mon Sep 30 14:35:10 2013
Copyright (c) 1982, 2013, Oracle. All rights reserved.
Connected to:
Oracle Database 11g Release 11.2.0.1.0 - 64bit Production
```

*The 64-bit sqlplus client was used to connect to a 64-bit 11g instance called MOAB, which is hosted on adaptive-oracle.ac*

5. Verify that you are logged in to the correct database.

```
SQL> select name from v$database
2 ;
NAME
-----
MOAB
```

6. Create the database in Oracle using the `moab-db-oracle-create.sh` script located in the `contrib/sql` directory in the root of the binaries.

**i** Useful comments are at the top of the script. Read the comments before running the script.

```
[root]# ./moab-db-oracle-create.sh
```

### 7. Display all of user MOAB's tables.

```
SQL> select table_name from all_tables where owner = 'MOAB';

TABLE_NAME
-----
TRIGGERS
MOAB
OBJECTTYPE
VCS
EVENTTYPE
JOBHISTORY
MCHECKPOINT
NODES
EVENTS
NODESTATSGENERICRESOURCES
JOBS
RESERVATIONS
GENERICMETRICS
REQUESTS
GENERALSTATS
NODESTATS

16 rows selected.
SQL>
```

### 8. Generate a script to describe all of user MOAB's tables. Cut and paste the following into a terminal that is *not* logged in to SQLPlus.

```
[root]# cat > /tmp/generateDescribe.sql <<EOL
SET HEADING OFF
SET FEEDBACK OFF
SET ECHO OFF
SET PAGESIZE 0
SPOOL /tmp/describeAllUserTables.sql
select 'desc '||owner||'.'||table_name||';' from all_tables where owner = 'MOAB';
SPOOL OFF
EOL
```

### 9. Run describeAllUserTables.sql.

```
[root]# SQL> start /tmp/describeAllUserTables.sql
```

## Related topics

- [Connecting to an Oracle Database with an ODBC Driver on page 799](#)
- [Database Configuration on page 793](#)

## Migrating Your Database to Newer Versions of Moab

Sometimes when upgrading from an older version of Moab to a newer version, you must update your database schema. If the schema Moab expects to operate against is different from the actual schema of the database Moab is connected to, Moab might not be able to use the database properly and data might be lost.

When upgrading the Moab database schema from an old version, you must perform each version upgrade in order. You cannot skip versions. For example, to migrate from version 6.1 to version 7.5, you must follow the steps in [Migrating from Moab 6.1 to Moab 7.0 on page 810](#), [Migrating from Moab 7.0 to](#)

[Moab 7.1 on page 810](#), [Migrating from Moab 7.1 to Moab 7.2 on page 810](#), [Migrating from Moab 7.2 to Moab 7.2.6 on page 809](#), then [Migrating from Moab 7.2.6 to Moab 7.5 on page 809](#), and so on.

**i** If you are upgrading your database to the 7.5 schema from 7.2.x where "x" is 5 or lower, you must complete the instructions for the following two sections in order:

- [Migrating from Moab 7.2 to Moab 7.2.6 on page 809](#)
- [Migrating from Moab 7.2.6 to Moab 7.5 on page 809](#).

## Migrating from Moab 7.5 to Moab 8.0

In Moab Workload Manager 8.0, column names that have become reserved words in newer versions of MySQL, PostgreSQL, and Oracle were renamed to eliminate the need to quote column names in SQL statements. Also, a few additional columns were added to existing tables to support Moab's Green feature. To upgrade your database with these changes, use the `moab-db-<database>-upgrade8_0.sql` file located in the `contrib/sql` directory in the root of the binaries. For example, to migrate your MySQL database from the 7.5 (or later) schema, run the following:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade8_0.sql
```

*The database name is usually "Moab".*

Similar migration scripts exist for Oracle and PostgreSQL.

## Migrating from Moab 7.2.6 to Moab 7.5

In Moab Workload Manager 7.5, column names that are reserved words in databases supported by Adaptive Computing were renamed to eliminate the need to quote column names in SQL statements. To upgrade your database with these changes, use the `moab-db-<database>-upgrade7_5.sql` file located in the `contrib/sql` directory in the root of the binaries. For example, to migrate your MySQL database from the 7.2.6 (or later) schema, run the following:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade7_5.sql
```

*The database name is usually "Moab".*

Similar migration scripts exist for Oracle and PostgreSQL.

## Migrating from Moab 7.2 to Moab 7.2.6

In Moab Workload Manager 7.2.6, several columns were extended and the primary key on the Triggers table changed. To upgrade your database with these changes, use the `moab-db-<database>-upgrade7_2_6.sql` file located in the `contrib/sql` directory in the root of the binaries. For example, to migrate your MySQL database from the 7.2.x (pre-7.2.6) schema to the 7.2.6 schema, run the following:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade7_2_6.sql
```

*The database name is usually "Moab".*

Similar migration scripts exist for Oracle and PostgreSQL.

**i** The 7.2.6 database upgrade is compatible with all earlier versions of 7.2.

## Migrating from Moab 7.1 to Moab 7.2

In Moab 7.2, several events in the event table related to the Accounting Manager were renamed. To upgrade your database with these changes, use the `moab-db-<database>-upgrade7_2.sql` file located in the `contrib/sql` directory in the root of the binaries. For example, to migrate your MySQL database from the 7.1 schema to the 7.2 schema, run the following:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade7_2.sql
```

*The database name is usually "Moab".*

Similar migration scripts exist for Oracle and PostgreSQL.

## Migrating from Moab 7.0 to Moab 7.1

In Moab 7.1, Offset was renamed TriggerOffset in the Triggers table. To upgrade your database with these changes, use the `moab-db-<database>-upgrade7_1.sql` file located in the root of the binaries. For example, to migrate your MySQL database from the 7.0 schema to the 7.1 schema, run the following:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade7_1.sql
```

*The database name is usually "Moab".*

Similar migration scripts exist for Oracle and PostgreSQL.

## Migrating from Moab 6.1 to Moab 7.0

In Moab 7.0, the Moab table has been removed from the database, and MoabInfo and JobHistory tables have been added to it. To upgrade your database with these changes, use the `moab-db-mysql-upgrade6_1.sql` file located in the `contrib/sql` directory in the root of the binaries. This is done by running the following command:

```
[root@]# mysql -u root -D <database name> -p < moab-db-mysql-upgrade6_1.sql
```

*The database name is usually "Moab".*

Your MySQL database is updated for Moab 7.0.

## Migrating from Moab 6.0 to Moab 6.1

An Events table has been added to the database in Moab 6.1. Update the `contrib/sql/moab-db.sql` file with the following table:

```
CREATE TABLE Events (
  ID INTEGER,
  ObjectType INTEGER,
  EventType INTEGER,
  EventTime INTEGER UNSIGNED,
  ObjectName VARCHAR(64),
  Name VARCHAR(64),
  Description TEXT,
  PRIMARY KEY (ID)
);
```

Use the `mdiag -e --xml` command in the following format to query the events table.

```
mdiag -e [-w <starttime>|<endtime>|<eventtypes>|<oidlist>|<eidlist>|<objectlist>] --xml
```

The table is then displayed with all specified events configured with the [RECORDEVENTLIST](#) parameter.

**i** If the command could return a large of data, redirect the output. `mdiag -e --xml > outputfile`

## Migrating from Moab 5.4 to Moab 6.0

The ODBC database schema has been updated for Moab 6.0. When updating Moab to version 6.0, the changes below must be applied to the database for database functionality to work. Below are the SQL statements required to update the schema for Moab 6.0.

**i** These changes are only necessary for an ODBC database. An SQLite database does not require an update.

```
ALTER TABLE Events ADD COLUMN Name VARCHAR(64);
ALTER TABLE Events MODIFY Description TEXT;
```

```
CREATE TABLE Nodes (
  ID VARCHAR(64),
  State VARCHAR(64),
  OperatingSystem VARCHAR(64),
  ConfiguredProcessors INTEGER UNSIGNED,
  AvailableProcessors INTEGER UNSIGNED,
  ConfiguredMemory INTEGER UNSIGNED,
  AvailableMemory INTEGER UNSIGNED,
  Architecture VARCHAR(64),
  AvailGres VARCHAR(64),
  ConfigGres VARCHAR(64),
  AvailClasses VARCHAR(64),
  ConfigClasses VARCHAR(64),
  ChargeRate DOUBLE,
  DynamicPriority DOUBLE,
  EnableProfiling INTEGER UNSIGNED,
  Features VARCHAR(64),
  GMetric VARCHAR(64),
  HopCount INTEGER UNSIGNED,
  HypervisorType VARCHAR(64),
  IsDeleted INTEGER UNSIGNED,
  IsDynamic INTEGER UNSIGNED,
  JobList VARCHAR(64),
  LastUpdateTime INTEGER UNSIGNED,
  LoadAvg DOUBLE,
  MaxLoad DOUBLE,
  MaxJob INTEGER UNSIGNED,
  MaxJobPerUser INTEGER UNSIGNED,
  MaxProc INTEGER UNSIGNED,
  MaxProcPerUser INTEGER UNSIGNED,
  OldMessages VARCHAR(64),
  NetworkAddress VARCHAR(64),
  NodeSubstate VARCHAR(64),
  Operations VARCHAR(64),
  OSList VARCHAR(64),
  Owner VARCHAR(64),
  ResOvercommitFactor VARCHAR(64),
  Partition VARCHAR(64),
  PowerIsEnabled INTEGER UNSIGNED,
  PowerPolicy VARCHAR(64),
  PowerSelectState VARCHAR(64),
  PowerState VARCHAR(64),
  Priority INTEGER UNSIGNED,
  PriorityFunction VARCHAR(64),
  ProcessorSpeed INTEGER UNSIGNED,
  ProvisioningData VARCHAR(64),
  AvailableDisk INTEGER UNSIGNED,
  AvailableSwap INTEGER UNSIGNED,
  ConfiguredDisk INTEGER UNSIGNED,
  ConfiguredSwap INTEGER UNSIGNED,
  ReservationCount INTEGER UNSIGNED,
  ReservationList VARCHAR(64),
  ResourceManagerList VARCHAR(64),
  Size INTEGER UNSIGNED,
  Speed DOUBLE,
  SpeedWeight DOUBLE,
  TotalNodeActiveTime INTEGER UNSIGNED,
  LastModifyTime INTEGER UNSIGNED,
  TotalTimeTracked INTEGER UNSIGNED,
```

```

    TotalNodeUpTime INTEGER UNSIGNED,
    TaskCount INTEGER UNSIGNED,
    VMOSList VARCHAR(64),
    PRIMARY KEY (ID)
);

CREATE TABLE Jobs (
  ID VARCHAR(64),
  SourceRMJobID VARCHAR(64),
  DestinationRMJobID VARCHAR(64),
  GridJobID VARCHAR(64),
  AName VARCHAR(64),
  User VARCHAR(64),
  Account VARCHAR(64),
  Class VARCHAR(64),
  QOS VARCHAR(64),
  OwnerGroup VARCHAR(64),
  JobGroup VARCHAR(64),
  State VARCHAR(64),
  EState VARCHAR(64),
  SubState VARCHAR(64),
  UserPriority INTEGER UNSIGNED,
  SystemPriority INTEGER UNSIGNED,
  CurrentStartPriority INTEGER UNSIGNED,
  RunPriority INTEGER UNSIGNED,
  PerPartitionPriority TEXT,
  SubmitTime INTEGER UNSIGNED,
  QueueTime INTEGER UNSIGNED,
  StartTime INTEGER UNSIGNED,
  CompletionTime INTEGER UNSIGNED,
  CompletionCode INTEGER,
  UsedWalltime INTEGER UNSIGNED,
  RequestedMinWalltime INTEGER UNSIGNED,
  RequestedMaxWalltime INTEGER UNSIGNED,
  CPULimit INTEGER UNSIGNED,
  SuspendTime INTEGER UNSIGNED,
  HoldTime INTEGER UNSIGNED,
  ProcessorCount INTEGER,
  RequestedNodes INTEGER,
  ActivePartition VARCHAR(64),
  SpecPAL VARCHAR(64),
  DestinationRM VARCHAR(64),
  SourceRM VARCHAR(64),
  Flags TEXT,
  MinPreemptTime INTEGER UNSIGNED,
  Dependencies TEXT,
  RequestedHostList TEXT,
  ExcludedHostList TEXT,
  MasterHostName VARCHAR(64),
  GenericAttributes TEXT,
  Holds TEXT,
  Cost DOUBLE,
  Description TEXT,
  Messages TEXT,
  NotificationAddress TEXT,
  StartCount INTEGER UNSIGNED,
  BypassCount INTEGER UNSIGNED,
  CommandFile TEXT,
  Arguments TEXT,
  RMSubmitLanguage TEXT,

```

```

StdIn TEXT,
StdOut TEXT,
StdErr TEXT,
RMOutput TEXT,
RMErrror TEXT,
InitialWorkingDirectory TEXT,
UMask INTEGER UNSIGNED,
RsvStartTime INTEGER UNSIGNED,
BlockReason TEXT,
BlockMsg TEXT,
PSDedicated DOUBLE,
PSUtilized DOUBLE,
PRIMARY KEY (ID)
);

CREATE TABLE Requests (
  JobID VARCHAR(64),
  RIndex INTEGER UNSIGNED,
  AllocNodeList VARCHAR(1024),
  AllocPartition VARCHAR(64),
  PartitionIndex INTEGER UNSIGNED,
  NodeAccessPolicy VARCHAR(64),
  PreferredFeatures TEXT,
  RequestedApp VARCHAR(64),
  RequestedArch VARCHAR(64),
  ReqOS VARCHAR(64),
  ReqNodeSet VARCHAR(64),
  ReqPartition VARCHAR(64),
  MinNodeCount INTEGER UNSIGNED,
  MinTaskCount INTEGER UNSIGNED,
  TaskCount INTEGER UNSIGNED,
  TasksPerNode INTEGER UNSIGNED,
  DiskPerTask INTEGER UNSIGNED,
  MemPerTask INTEGER UNSIGNED,
  ProcsPerTask INTEGER UNSIGNED,
  SwapPerTask INTEGER UNSIGNED,
  NodeDisk INTEGER UNSIGNED,
  NodeFeatures TEXT,
  NodeMemory INTEGER UNSIGNED,
  NodeSwap INTEGER UNSIGNED,
  NodeProcs INTEGER UNSIGNED,
  GenericResources TEXT,
  ConfiguredGenericResources TEXT,
  PRIMARY KEY (JobID,RIndex)
);

INSERT INTO ObjectType (Name, ID) VALUES ("Rsv", 13);
INSERT INTO ObjectType (Name, ID) VALUES ("RM", 14);
INSERT INTO ObjectType (Name, ID) VALUES ("Sched", 15);
INSERT INTO ObjectType (Name, ID) VALUES ("SRsv", 16);
INSERT INTO ObjectType (Name, ID) VALUES ("Sys", 17);
INSERT INTO ObjectType (Name, ID) VALUES ("TNode", 18);
INSERT INTO ObjectType (Name, ID) VALUES ("Trig", 19);
INSERT INTO ObjectType (Name, ID) VALUES ("User", 20);
INSERT INTO ObjectType (Name, ID) VALUES ("CJob", 23);
INSERT INTO ObjectType (Name, ID) VALUES ("GRes", 30);
INSERT INTO ObjectType (Name, ID) VALUES ("Gmetric", 31);
INSERT INTO ObjectType (Name, ID) VALUES ("Stats", 39);
INSERT INTO ObjectType (Name, ID) VALUES ("TJob", 42);
INSERT INTO ObjectType (Name, ID) VALUES ("Paction", 43);
INSERT INTO ObjectType (Name, ID) VALUES ("VM", 45);
INSERT INTO ObjectType (Name, ID) VALUES ("JGroup", 48);

```

```

INSERT INTO EventType (Name, ID) VALUES ("TRIGTHRESHOLD", 41);
INSERT INTO EventType (Name, ID) VALUES ("VMCREATE", 42);
INSERT INTO EventType (Name, ID) VALUES ("VMDESTROY", 43);
INSERT INTO EventType (Name, ID) VALUES ("VMMIGRATE", 44);
INSERT INTO EventType (Name, ID) VALUES ("VMPOWERON", 45);
INSERT INTO EventType (Name, ID) VALUES ("VMPOWEROFF", 46);
INSERT INTO EventType (Name, ID) VALUES ("NODEMODIFY", 47);
INSERT INTO EventType (Name, ID) VALUES ("NODEPOWEROFF", 48);
INSERT INTO EventType (Name, ID) VALUES ("NODEPOWERON", 49);
INSERT INTO EventType (Name, ID) VALUES ("NODEPROVISION", 50);
INSERT INTO EventType (Name, ID) VALUES ("ALLSCHEDCOMMAND", 51);
INSERT INTO EventType (Name, ID) VALUES ("AMCANCEL", 52);
INSERT INTO EventType (Name, ID) VALUES ("AMDEBIT", 53);
INSERT INTO EventType (Name, ID) VALUES ("AMQUOTE", 54);
INSERT INTO EventType (Name, ID) VALUES ("AMRESERVE", 55);
INSERT INTO EventType (Name, ID) VALUES ("RMPOLLEND", 56);
INSERT INTO EventType (Name, ID) VALUES ("RMPOLLSTART", 57);
INSERT INTO EventType (Name, ID) VALUES ("SCHEDCYCLEEND", 58);
INSERT INTO EventType (Name, ID) VALUES ("SCHEDCYCLESTART", 59);
INSERT INTO EventType (Name, ID) VALUES ("JOBCHECKPOINT", 60);

ALTER TABLE GeneralStats ADD COLUMN TotalConfiguredProcCount INTEGER;

```

## Importing Statistics from stats/DAY.\* to the Moab Database

The contrib/stat\_converter folder contains the files to build mstat\_converter, an executable that reads file-based statistics in a Moab stats directory and dumps them into a database. It also reads the Moab checkpoint file (.moab.ck) and dumps that to the database as well. It uses the \$MOABHOMEDIR/moab.cfg file to connect to the appropriate database and reads the statistics files from \$MOABHOMEDIR/stats.

To run, execute the program mstat\_converter with no arguments.

The statistics converter program does not clear the database before converting. However, if there are statistics in the database and the statistics files from the same period, the converter overwrites the database information with the information from the statistics files.

## Accelerators

Moab can integrate with the TORQUE resource manager to discover, report, schedule, and submit workload to various accelerator architectures (such as NVIDIA GPUs or Intel® Xeon Phi™ co-processor architecture) for parallel processing. See the topics below for specific information.

- [Scheduling GPUs](#)
  - [Using GPUs with NUMA](#)
  - [NVIDIA GPUs](#)

- [GPU Metrics](#)
- [Configuring Intel® Xeon Phi™ Co-processor Architecture](#)
  - [Intel® Xeon Phi™ Co-processor Metrics](#)

## Scheduling GPUs

In TORQUE 2.5.4 and later, users can request GPUs on a node at job submission by specifying a nodes resource request, using the `qsub -l` option. The number of GPUs a node has must be specified in the nodes file. The GPU is then reported in the output of `pbsnodes`:

```
napali
state = free
np = 2
ntype = cluster
status = rectime=1288888871,varattr=,jobs=,state=free,netload=1606207294,gres=tom:!/home/dbeer/dev/scripts/dynamic_
resc.sh,loadave=0.10,ncpus=2,physmem=3091140kb,availmem=32788032348kb,
totmem=34653576492kb,idletime=4983,nusers=3,nsessions=14,sessions=3136 1805 2380 2428
1161 3174 3184
3191 3209 3228 3272 3333 20560 32371,uname=Linux napali 2.6.32-25-generic #45-Ubuntu
SMP Sat Oct 16 19:52:42
UTC 2010 x86_64,opsys=linux
mom_service_port = 15002
mom_manager_port = 15003
gpus = 1
```

The `$PBS_GPUFILE` has been created to include GPU awareness. The GPU appears as a separate line in `$PBS_GPUFILE` and follows this syntax:

```
<hostname>-gpu<index>
```

If a job were submitted to run on a server called "napali" (the submit command would look something like: `qsub test.sh -l nodes=1:ppn=2:gpus=1`), the `$PBS_GPUFILE` would contain:

```
napali-gpu0
```

It is left up to the job's owner to make sure that the job executes properly on the GPU. By default, TORQUE treats GPUs exactly the same as `ppn` (which corresponds to CPUs).

### Related topics

- [Using GPUs with NUMA](#)
- [NVIDIA GPUs](#)

## Using GPUs with NUMA

The `pbs_server` requires awareness of how the MOM is reporting nodes since there is only one MOM daemon and multiple MOM nodes. Configure the `server_priv/nodes` file with the `num_node_boards` and `numa_gpu_node_str` attributes. The attribute `num_node_boards` tells `pbs_server` how many NUMA nodes are reported by the MOM. If each NUMA node has the same number of GPUs, add the total

number of GPUs to the nodes file. Following is an example of how to configure the nodes file with `num_node_boards`:

```
numahost gpus=12 num_node_boards=6
```

This line in the nodes file tells `pbs_server` there is a host named `numahost` and that it has 12 GPUs and 6 nodes. The `pbs_server` divides the value of GPUs (12) by the value for `num_node_boards` (6) and determines there are 2 GPUs per NUMA node.

In this example, the NUMA system is uniform in its configuration of GPUs per node board, but a system does not have to be configured with the same number of GPUs per node board. For systems with non-uniform GPU distributions, use the attribute `numa_gpu_node_str` to let `pbs_server` know where GPUs are located in the cluster.

If there are equal numbers of GPUs on each NUMA node, you can specify them with a string. For example, if there are 3 NUMA nodes and the first has 0 GPUs, the second has 3, and the third has 5, you would add this to the nodes file entry:

```
numa_gpu_node_str=0,3,5
```

In this configuration, `pbs_server` knows it has three MOM nodes and the nodes have 0, 3s, and 5 GPUs respectively. Note that the attribute `gpus` is not used. The `gpus` attribute is ignored because the number of GPUs per node is specifically given.

In TORQUE 3.0.2 or later, `qsub` supports the mapping of `-l gpus=X` to `-l gres=gpus:X`. This allows users who are using NUMA systems to make requests such as `-l ncpus=20,gpus=5` ( or `-l ncpus=20:gpus=5`) indicating they are not concerned with the GPUs in relation to the NUMA nodes they request; they only want a total of 20 cores and 5 GPUs.

### Related topics

- [Scheduling GPUs](#)
- [NVIDIA GPUs](#)

## NVIDIA GPUs

The `pbs_mom` file can now query for GPU hardware information and report status to the `pbs_server`. `gpustatus` will appear in `pbsnodes` output. New commands allow for setting GPU modes and for resetting GPU ECC error counts.

 This feature is only available in TORQUE 2.5.6, 3.0.2, and later.

 This document assumes that you have installed the NVIDIA CUDA ToolKit and the NVIDIA development drivers on a compute node with an NVIDIA GPU. (Both can be downloaded from <http://developer.nvidia.com/category/zone/cuda-zone>).

You will want to download the latest version if you run into problems compiling.

If the `pbs_server` does not have GPUs, it only needs to be configured with `--enable-nvidia-gpus`. All other systems that have NVIDIA GPUs will need:

- `--enable-nvidia-gpus`
- `--with-nvml-include=DIR` (include path for `nvml.h`)

 `nvml.h` is only found in the NVIDIA CUDA ToolKit.

- `--with-nvml-lib=DIR` (\*lib path for `libnvidia-ml`)

Systems that have NVIDIA GPUs require the following:

Server

```
./configure --with-debug --enable-nvidia-gpus
```

Compute nodes (with NVIDIA GPUs)

```
./configure --with-debug --enable-nvidia-gpus --with-nvml-lib=/usr/lib64 --with-nvml-include=/cuda/NVML
```

If all of the compute nodes have the same hardware and software configuration, you can choose to compile on one compute node and then run `make packages`.

```
> make packages
Building ./torque-package-clients-linux-x86_64.sh ...
Building ./torque-package-mom-linux-x86_64.sh ...
Building ./torque-package-server-linux-x86_64.sh ...
Building ./torque-package-gui-linux-x86_64.sh ...
Building ./torque-package-devel-linux-x86_64.sh ...
Done.
```

The package files are self-extracting packages that can be copied and executed on your production machines. (Use `--help` for options.)

 When updating, it is good practice to stop the `pbs_server` and make a backup of the TORQUE home directory. You will also want to backup the output of `qmgr -c 'p s'`. The update will only overwrite the binaries.

 If you move GPU cards to different slots, you must restart `pbs_server` in order for TORQUE to recognize the drivers as the same ones in different locations rather than 2 new, additional drivers.

For further details, see these topics:

- [TORQUE configuration on page 818](#)
- [GPU modes for NVIDIA 260.x driver on page 819](#)
- [GPU Modes for NVIDIA 270.x driver on page 819](#)
- [gpu\\_status on page 820](#)
- [New NVIDIA GPU support on page 820](#)

## TORQUE configuration

There are three configuration (`./configure`) options available for use with Nvidia GPGPUs:

- `--enable-nvidia-gpus`
- `--with-nvml-lib=DIR`
- `--with-nvml-include=DIR`

`--enable-nvidia-gpus` is used to enable the new features for the Nvidia GPGPUs. By default, the `pbs_moms` use the `nvidia_smi` command to interface with the Nvidia GPUs.

```
./configure --enable-nvidia-gpus
```

To use the NVML (NVIDIA Management Library) API instead of `nvidia-smi`, configure TORQUE using `--with-nvml-lib=DIR` and `--with-nvml-include=DIR`. These commands specify the location of the `libnvidia-ml` library and the location of the `nvml.h` include file.

```
./configure --with-nvml-lib=/usr/lib
--with-nvml-include=/usr/local/cuda/Tools/NVML
server_priv/nodes:
node001 gpus=1
node002 gpus=4
...
pbsnodes -a
node001
...
    gpus = 1
...
```

By default, when TORQUE is configured with `--enable-nvidia-gpus` the `$TORQUE_HOME/nodes` file is automatically updated with the correct GPU count for each MOM node. See the TORQUE documentation on [ggpumode](#) for additional information.

## GPU modes for NVIDIA 260.x driver

- 0 – Default - Shared mode available for multiple processes
- 1 – Exclusive - Only one COMPUTE thread is allowed to run on the GPU
- 2 – Prohibited - No COMPUTE contexts are allowed to run on the GPU

## GPU Modes for NVIDIA 270.x driver

- 0 – Default - Shared mode available for multiple processes
- 1 – Exclusive Thread - Only one COMPUTE thread is allowed to run on the GPU (v260 exclusive)
- 2 – Prohibited - No COMPUTE contexts are allowed to run on the GPU
- 3 – Exclusive Process - Only one COMPUTE process is allowed to run on the GPU

## gpu\_status

```

root@gpu:~# pbsnodes gpu
gpu
...
  gpus = 2
  gpu status = gpu[1]=gpu_id=0:6:0;gpu_product_name=Tesla
  C2050;gpu_display=Disabled;gpu_pci_device_id=6D110DE;gpu_pci_location_id=0:6:0;
  gpu_fan_speed=54 %;gpu_memory_total=2687 Mb;gpu_memory_used=74
  Mb;gpu_mode=Default;gpu_state=Unallocated;gpu_utilization=96
  %;gpu_memory_utilization=10
  %;gpu_ecc_mode=Enabled;gpu_single_bit_ecc_errors=0;gpu_double_bit_ecc_errors=
  0;gpu_temperature=88 C,gpu[0]=gpu_id=0:5:0;gpu_product_name=Tesla
  C2050;gpu_display=Enabled;gpu_pci_device_id=6D110DE;gpu_pci_location_id=0:5:0;
  gpu_fan_speed=66 %;gpu_memory_total=2687 Mb;gpu_memory_used=136
  Mb;gpu_mode=Default;gpu_state=Unallocated;gpu_utilization=96
  %;gpu_memory_utilization=10
  %;gpu_ecc_mode=Enabled;gpu_single_bit_ecc_errors=0;
  gpu_double_bit_ecc_errors=0;gpu_temperature=86 C,driver_ver=270.41.06,timestamp=Wed
  May 4 13:00:35
  2011

```

## New NVIDIA GPU support

qsub allows specifying required compute mode when requesting GPUs. If no GPU mode is requested, it will default to "exclusive" for Nvidia driver version 260 or "exclusive\_thread" for NVIDIA driver version 270 and above.

- qsub -l nodes=1:ppn=1:gpus=1
- qsub -l nodes=1:gpus=1
- qsub -l nodes=1:gpus=1:exclusive\_thread
- qsub -l nodes=1:gpus=1:exclusive\_process
- qsub -l nodes=1:gpus=1:reseterr
- qsub -l nodes=1:gpus=1:reseterr:exclusive\_thread (exclusive\_thread:reseterr)
- qsub -l nodes=1:gpus=1:reseterr:exclusive\_process

### Related topics

- [Scheduling GPUs on page 816](#)
- [Using GPUs with NUMA on page 816](#)

## GPU Metrics

GPU metrics can be collected for nodes that:

- Have one or more GPUs.
- Run TORQUE 2.5.x or later.
- Use NVIDIA drivers v260.x or v270.x.

GPU metric tracking must be enabled in `moab.cfg`:

```
RMCFG[torque] flags=RECORDGPUMETRICS
```

**i** There is one GPU metric for all GPU devices within a node (`gpu_timestamp`) and nine GPU metrics for each GPU device within a node. If the maximum GPU devices within a node is 4, you must increase the `MAXGMETRIC` value in `moab.cfg` by  $(\text{maxgpudevices} \times \text{gpumetrics}) + 1$ . In this case, the formula is  $(4 \times 9) + 1 = 37$ , so whatever the `MAXGMETRIC` value is, it must be increased by 37. This way, when enabling GPU metrics recording, Moab has enough GMETRIC types to accommodate the GPU metrics.

## GPU Metrics Map

The GPU metric names map is as follows (where *X* is the GPU number):

Metric name as returned by pbsnodes	GMETRIC name as stored in Moab	Metric output
<b>timestamp</b>	gpu_timestamp <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> <b>i</b> The <code>gpu_timestamp</code> metric is global to all GPUs on the node and indicates the last time the driver collected information on the GPUs.           </div>	The time data was collected in epoch time
<b>gpu_fan_speed</b>	gpuX_fan	The current fan speed as a percentage
<b>gpu_memory_total</b>	gpuX_mem	The total GPU memory in megabytes
<b>gpu_memory_used</b>	gpuX_usedmem	The total used GPU memory in megabytes
<b>gpu_utilization</b>	gpuX_util	The GPU capability currently in use as a percentage
<b>gpu_memory_utilization</b>	gpuX_memutil	The GPU memory currently in use as a percentage
<b>gpu_ecc_mode</b>	gpuX_ecc	Whether ECC is enabled or disabled
<b>gpu_single_bit_ecc_errors</b>	gpuX_ecc1err	The total number of EEC single-bit errors since the last counter reset

Metric name as returned by pbsnodes	GMETRIC name as stored in Moab	Metric output
<b>gpu_double_bit_ecc_errors</b>	gpuX_ecc2err	The total number of EEC double-bit errors since the last counter reset
<b>gpu_temperature</b>	gpuX_temp	The GPU current temperature in Celsius

Example 3-157: GPU example

```

$ mdiag -n -v --xml

<Data>
<node AGRES="GPUS=2;"
AVLCLASS="[test 8][batch 8]"
CFGCLASS="[test 8][batch 8]"
GMETRIC="gpu1_fan:59.00,gpu1_mem:2687.00,gpu1_usedmem:74.00,gpu1_util:94.00,gpu1_
memutil:9.00,gpu1_ecc:0.00,gpu1_ecc1err:0.00,gpu1_ecc2err:0.00,gpu1_temp:89.00,gpu0_
fan:70.00,gpu0_mem:2687.00,gpu0_usedmem:136.00,gpu0_util:94.00,gpu0_memutil:9.00,gpu0_
ecc:0.00,gpu0_ecc1err:0.00,gpu0_ecc2err:0.00,gpu0_temp:89.00,gpu_
timestamp:1304526680.00"
GRES="GPUS=2;"
LASTUPDATETIME="1304526518" LOAD="1.050000"
MAXJOB="0" MAXJOBPERUSER="0" MAXLOAD="0.000000" NODEID="gpu"
NODEINDEX="0" NODESTATE="Idle" OS="linux" OSLIST="linux"
PARTITION="makai" PRIORITY="0" PROCSPEED="0" RADISK="1"
RAMEM="5978" RAPROC="7" RASWAP="22722" RCDISK="1" RCMEM="5978"
RCPROC="8" RCSWAP="23493" RMACCESSLIST="makai" SPEED="1.000000"
STATMODIFYTIME="1304525679" STATTOTALTIME="315649"
STATUPTIME="315649"></node>
</Data>

```

## Intel® Xeon Phi™ Coprocessor Configuration

### Intel Many-Integrated Cores (MIC) architecture configuration

If you use an Intel Many-Integrated Cores (MIC) architecture-based product (e.g., Intel Xeon Phi™) in your cluster for parallel processing, you must configure TORQUE to detect them.

#### Prerequisites

- TORQUE 4.2 or later
- If you set up TORQUE using auto-detection and intend to get the MIC-based device status report, you must build pbs\_mom on a system that has the lower-level API libraries for the MIC-based device(s) installed. Additionally, every MOM built with `--enable-mics` and running on a compute node must already have the lower-level API libraries installed on the node. Note that the library is called `coi_host`. You must obtain the API libraries from [Intel](#).

## Setup Options

There are two ways to configure MIC-based devices with TORQUE: (1) manually and (2) by auto-detection.

### Manual configuration

- Add `mics=X` to the [nodes file](#) for the appropriate nodes.

```
napali np=12 mics=2
```

### Auto-detect

When you use auto-detection, `pbs_mom` discovers the MIC-based devices and reports them to `pbs_server`.

- At build time, add `--enable-mics` to the configure line.

```
./configure --enable-mics <other configure options>
```

## Validating the configuration

### TORQUE

#### *pbsnodes*

#### Example 3-158: *pbsnodes* output

```
slesmic
  state = free
  np = 100
  ntype = cluster
  status =
rectime=1347634381,varattr=,jobs=,state=free,netload=7442004852,gres=,loadave=0.00,ncp
us=32,physmem=65925692kb,availmem=66531344kb,totmem=68028984kb,idletime=59059,nusers=2
,nsessions=8,sessions=4387 4391 4392 4436 4439 4443 4459 100395,uname=Linux slesmic
3.0.13-0.27-default #1 SMP Wed Feb 15 13:33:49 UTC 2012 (d73692b) x86_64,opsys=linux
  mom_service_port = 15002
  mom_manager_port = 15003
  mics = 2
  mic_status = mic[1]=mic_id=8796;num_cores=61;num_threads=244;physmem=8065748992;free_
physmem=7854972928;swap=0;free_swap=0;max_frequency=1090;isa=COI_ISA_
KNC;load=0.000000;normalized_load=0.000000;,mic[0]=mic_id=8796;num_cores=61;num_
threads=244;physmem=8065748992;free_physmem=7872712704;swap=0;free_swap=0;max_
frequency=1090;isa=COI_ISA_KNC;load=0.540000;normalized_load=0.008852;

rhmic.ac
  state = free
  np = 100
  ntype = cluster
  status =
rectime=1347634381,varattr=,jobs=,state=free,netload=3006171583,gres=,loadave=0.00,ncp
us=32,physmem=65918268kb,availmem=66901588kb,totmem=67982644kb,idletime=59477,nusers=2
,nsessions=2,sessions=3401 29320,uname=Linux rhmic.ac 2.6.32-220.el6.x86_64 #1 SMP Tue
Dec 6 19:48:22 GMT 2011 x86_64,opsys=linux
  mom_service_port = 15002
  mom_manager_port = 15003
  mics = 1
  mic_status = mic[0]=mic_id=8796;num_cores=61;num_threads=244;physmem=8065748992;free_
physmem=7872032768;swap=0;free_swap=0;max_frequency=1090;isa=COI_ISA_
KNC;load=0.540000;normalized_load=0.008852;<mic_status>;
```

## Moab

### *mdiag -n -v*

#### Example 3-159: *mdiag -n -v* output

```

$ mdiag -n -v
compute node summary
Name           State   Procs      Memory      Disk      Swap
Speed  Ophys  Arch Par   Load Classes
hola                Idle    4:4      8002:8002    1:1      10236:13723
1.00  linux    - hol   0.24 [batch]
GRES=MICS:2,
-----
                ---    4:4      8002:8002    1:1      10236:13723

Total Nodes: 1 (Active: 0 Idle: 1 Down: 0)

```

### *checknode -v*

#### Example 3-160: *checknode* output

```

$ checknode slesmic
node slesmic

State:      Idle (in current state for 00:00:16)
Configured Resources: PROCS: 100 MEM: 62G SWAP: 64G DISK: 1M MICS: 2
Utilized Resources: SWAP: 1581M
Dedicated Resources: ---
Generic Metrics:   mic1_mic_id=8796.00,mic1_num_cores=61.00,mic1_num
threads=244.00,mic1_physmem=8065748992.00,mic1_free_physmem=7854972928.00,mic1_
swap=0.00,mic1_free_swap=0.00,mic1_max_frequency=1090.00,mic1_load=0.12,mic1_
normalized_load=0.00,mic0_mic_id=8796.00,mic0_num_cores=61.00,mic0_num
threads=244.00,mic0_physmem=8065748992.00,mic0_free_physmem=7872679936.00,mic0_
swap=0.00,mic0_free_swap=0.00,mic0_max_frequency=1090.00
  MTBF(longterm):  INFINITY MTBF(24h):  INFINITY
Opsys:      linux      Arch:      ---
Speed:      1.00      CPUload:   0.000
Classes:    [batch]
RM[napali]* TYPE=PBS
EffNodeAccessPolicy: SHARED

Total Time: 3:45:43 Up: 3:45:43 (100.00%) Active: 00:00:00 (0.00%)

Reservations:
  ---

```

## Job submission

### Syntax

Example 3-161: Request MIC-based device(s) in qsub

```
qsub .... -l nodes=X:mics=Y
```

*Because these resources are delimited with a colon, this command requests a job with X nodes and Y mics per task. If you run the same command and delimit the resources with a comma (qsub .... -l nodes=X,mics=Y), you request a job with X nodes and Y mics per job.*

### qstat -f

Example 3-162: qstat -f output

```
Job Id: 5271.napali
Job_Name = STDIN
Job_Owner = dbeer@napali
job_state = Q
queue = batch
server = napali
Checkpoint = u
ctime = Fri Sep 14 08:56:33 2012
Error_Path = napali:/home/dbeer/dev/private-torque/trunk/STDIN.e5271
Hold_Types = n
Join_Path = oe
Keep_Files = n
Mail_Points = a
mtime = Fri Sep 14 08:56:33 2012
Output_Path = napali:/home/dbeer/dev/private-torque/trunk/STDIN.o5271
Priority = 0
qtime = Fri Sep 14 08:56:33 2012
Rerunable = True
Resource_List.nodect = 1
Resource_List.nodes = 1:mics=1
substate = 10
Variable_List = PBS_O_QUEUE=batch,PBS_O_HOME=/home/dbeer,
                PBS_O_LOGNAME=dbeer,
                PBS_O_PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/b
                in:/usr/games,PBS_O_MAIL=/var/mail/dbeer,PBS_O_SHELL=/bin/bash,
                PBS_O_LANG=en_US.UTF-8,
                PBS_O_SUBMIT_FILTER=/usr/local/sbin/torque_submitfilter,
                PBS_O_WORKDIR=/home/dbeer/dev/private-torque/trunk,PBS_O_HOST=napali,
                PBS_O_SERVER=napali
euser = dbeer
egroup = company
queue_rank = 3
queue_type = E
etime = Fri Sep 14 08:56:33 2012
submit_args = -l nodes=1:mics=1
fault_tolerant = False
job_radix = 0
submit_host = napali
```

## checkjob -v

Example 3-163: checkjob -v output

```

dthompson@mahalo:~/dev/moab-test/trunk$ checkjob -v 2
job 2 (RM job '2.mahalo')

AName: STDIN
State: Idle
Creds: user:dthompson group:dthompson class:batch
WallTime: 00:00:00 of 1:00:00
SubmitTime: Thu Sep 13 17:06:06
(Time Queued Total: 00:00:24 Eligible: 00:00:02)

TemplateSets: DEFAULT
Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: ALL
Dedicated Resources Per Task: PROCS: 1 MICS: 1

...

```

## Intel® Xeon Phi™ Co-processor Metrics

Intel Many-Integrated Cores (MIC) architecture-based device (e.g., Intel Xeon Phi™) metrics can be collected for nodes that:

- Have one or more MIC-based devices.
- Run TORQUE 4.2.x or later.
- Run Moab 7.2 or later.

MIC-based device metric tracking must be enabled in `moab.cfg`:

```

RMCFG[torque] flags=RECORDMICMETRICS

```

**i** There are 11 metrics for each MIC-based device within a node. If the maximum MIC-based devices within a node is 4, you must increase the `MAXGMETRIC` value in `moab.cfg` by  $(maxmicdevices \times micmetrics)$ . In this case, the formula is  $(4 \times 11) = 44$ , so whatever the `MAXGMETRIC` value is, it must be increased by 44. This way, when enabling MIC-based device metrics recording, Moab has enough GMETRIC types to accommodate the additional metrics.

## MIC-based Metrics Map

The MIC-based metric names map is as follows (where *X* is the MIC-based device number):

Metric name as returned by pbsnodes	GMETRIC name as stored in Moab	Metric output
mic_id	micX_mic_id	The ID of the MIC-based device
num_cores	micX_num_cores	The number of cores in the MIC-based device
num_threads	micX_num_threads	The number of hardware threads on the MIC-based device
physmem	micX_physmem	The total physical memory in the MIC-based device
free_physmem	micX_free_physmem	The available physical memory in the MIC-based device
swap	micX_swap	The total swap space on the MIC-based device
free_swap	micX_free_swap	The unused swap space on the MIC-based device
max_frequency	micX_max_frequency	The maximum frequency speed of any core in the MIC-based device
isa	micX_isa	The hardware interface type of the MIC-based device
load	micX_load	The total current load of the MIC-based device
normalized_load	micX_normalized_load	The normalized load of the MIC-based device (total load divided by number of cores in the MIC-based device)

## Preemption

### About preemption

Sites possess workloads of varying importance, and users may want to run jobs with higher priorities before jobs with lower priorities. This can be done by using preemption. Preemption is simply the process by which a higher-priority job can take the place of a lower-priority job. You can also use preemption for optimistic scheduling and development job support.

This section explains how to configure and use preemption. [Simple example of preemption on page 848](#) offers a basic introduction and contains examples to help you get started using preemption. The other sections offer more explanation and information about what you can do with preemption and contain some best practices that will help you avoid the need for troubleshooting in the future.

While this section does not explain every possible preemption configuration, it does prescribe the best practices for setting up and using preemption with your system. It is recommended that you follow the established instructions contained in this section.

 Preemption does not work with dynamic provisioning.

 Neither *SPANEVENLY* nor *DELAY* values of the `NODESETPLUS` parameter will work with multi-req jobs or preemption.

 Do not allow preemption with interactive jobs unless `PREEMTPOLICY` is set to *CANCEL*. (For more information, see [Canceling jobs with preemption on page 830](#).)

Tasks associated with preemption:

The following sections include information about each type of preemption, their different usage benefits, and any configurations and settings needed to use them.

- [Canceling jobs with preemption on page 830](#)
- [Checkpointing jobs with preemption on page 833](#)
- [Requeueing jobs with preemption on page 834](#)
- [Suspending jobs with preemption on page 837](#)
- [Using owner preemption on page 840](#)
- [Using QoS preemption on page 844](#)

Preemption references:

These sections contain information that you can use as references for the preemption tasks.

- [Manual preemption commands on page 845](#)
- [Preemption flags on page 846](#)
- [PREEMTPOLICY types on page 848](#)
- [Simple example of preemption on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

Related topics

- [Optimizing Scheduling Behavior - Backfill and Node Sets on page 507](#)

## How-to's

### Canceling jobs with preemption

#### Context

**CANCEL** is one of the [PREEMTPOLICY](#) types (for more information, see [PREEMTPOLICY types on page 848](#)). The **CANCEL** attribute cancels active jobs, regardless of any **JOBFLAGS** (such as **REQUEUEABLE** or **SUSPENDABLE**). (For more information, see [Job Flags on page 145](#).)

For information about **PREEMPTEE** and **PREEMPTOR** flags, see [Preemption flags on page 846](#)

**i** You should not allow preemption with interactive jobs unless **PREEMTPOLICY** is set to **CANCEL**.

The following outlines some benefits of using **CANCEL** and also lists some things you should be aware of if you choose to use it.

#### Advantages:

This attribute is the easiest to configure and use.

#### Cautions:

Canceled jobs are not automatically restarted or requeued. Users must resubmit canceled jobs.

### To preempt jobs using CANCEL

1. Make the following configurations to the `moab.cfg` file:
  - a. Set **GUARANTEEDPREEMPTION** to **TRUE**. (This causes Moab to lock **PREEMPTOR** jobs until **JOBRETRYTIME** expires.)
  - b. Make sure that **JOBNODEMATCHPOLICY** is *not* set to **EXACTNODE**, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
  - c. Set **PREEMTPOLICY** to **CANCEL** (for more information, see [PREEMTPOLICY types on page 848](#)).
  - d. Make sure that the **PREEMPTEE** job has a lower priority than the **PREEMPTOR** job (for more information, see [Preemption flags on page 846](#)).

For example:

```
GUARANTEEDPREEMPTION TRUE
PREEMTPOLICY CANCEL

QOSCFG[test1] QFLAGS=PREEMPTEE MEMBERULIST=john PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=10000
```

2. Submit a job to the preemptee QoS (`test1`). For example:

```
[john@g06]# echo sleep 600 | msub -l walltime=600 -l qos=test1 -l procs=128
```

(Optional) Examine the following output for showq:

```
Moab.7
[john@g06]# showq

active jobs-----
JOBID      USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.7     john       Running    128        00:01:59    Thu Nov 10 12:28:44

1 active job      128 of 128 processors in use by local jobs (100.00%)
                    2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 eligible jobs

blocked jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 blocked jobs

Total job: 1
```

3. Now submit a job to the preemptor QoS (test2). For example:

```
[john@g06]$ echo sleep 120 | msub -l procs=128,walltime=120 -l qos=test2
```

(Optional) Examine the following output for showq:

```
Moab.8
[john@g06]# showq

active jobs-----
JOBID      USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.7     john       Canceled  128        00:01:56    Thu Nov 10 12:28:44
Moab.8     john       Running    128        00:02:00    Thu Nov 10 12:28:48

2 active jobs 128 of 128 processors in use by local jobs (100.00%)
                    2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 eligible jobs

blocked jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 blocked jobs

Total jobs: 2
```

*Note that test1 is canceled when test2 is submitted.*

(Optional) Examine the checkjob outputs for these two jobs:

```
[john@g06]$ checkjob Moab.9
job Moab.9

State: Removed
Completion Code: -1 Time: Thu Nov 10 12:28:48
Creds: user:john group:john qos:test1
WallTime: 00:00:02 of 00:02:00
SubmitTime: Thu Nov 10 12:28:44
(Time Queued Total: 00:00:07 Eligible: 00:00:00)

Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses
NodeCount: 2

Allocated Nodes:
node[01-02]*64

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.UFe8sQ

StartCount: 1
Flags: GLOBALQUEUE, PROCSPECIFIED
Attr: PREEMPTTEE
StartPriority: 100
```

*Note that the preempted job has been removed.*

```
[john@g06]$ checkjob Moab.10
job Moab.10

State: Running
Creds: user:john group:john qos:test2
WallTime: 00:00:00 of 00:02:00
SubmitTime: Thu Nov 10 12:36:31
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Thu Nov 10 12:28:48
Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses

Allocated Nodes:
node[01-02]*64

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.CZavjU

StartCount: 1
Flags: HASPREEMPTED, PREEMPTOR, GLOBALQUEUE, PROCSPECIFIED
StartPriority: 10000
Reservation 'Moab.10' (-00:00:07 -> 00:01:53 Duration: 00:02:00)
```

## Related topics

- [Suspending jobs with preemption on page 837](#)
- [Checking jobs with preemption on page 833](#)

- [Requeueing jobs with preemption on page 834](#)
- [Preemption flags on page 846](#)
- [About preemption on page 828](#)
- [PREEMTPOLICY types on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

## Checkpointing jobs with preemption

### Context

*CHECKPOINT* is one of the [PREEMTPOLICY](#) types (for more information, see [PREEMTPOLICY types on page 848](#)). For systems that allow checkpointing, the *CHECKPOINT* attribute allows a job to save its current state and either terminate or continue running. A checkpointed job may restart at any time and resume execution from its most recent checkpoint.

You can tune checkpointing behavior on a per-resource manager-basis by setting the *CHECKPOINTSIG* and *CHECKPOINTTIMEOUT* attributes of the [RMCFG](#) parameter.

For information about *PREEMPTTEE* and *PREEMPTOR* flags, see [Preemption flags on page 846](#)

The following outlines some benefits of using *CHECKPOINT* and also lists some things you should be aware of if you choose to use it.

#### Advantages:

This attribute allows you to restart a job from its last checkpoint.

#### Cautions:

Jobs tend to take longer to complete when you use *CHECKPOINT*.

## To preempt jobs using CHECKPOINT

Make the following configurations to the `moab.cfg` file:

1. Set [GUARANTEEDPREEMPTION](#) to *TRUE*. (This causes Moab to lock *PREEMPTOR* jobs until [JOBRETRYTIME](#) expires.) (This locks the job on a node and keeps trying to preempt.)
2. Make sure that [JOBNODEMATCHPOLICY](#) is *not* set to *EXACTNODE*, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
3. Set [PREEMTPOLICY](#) to *CHECKPOINT* (for more information, see [PREEMTPOLICY types on page 848](#)).
4. Make sure that the *PREEMPTTEE* job has a lower priority than the *PREEMPTOR* job (for more information, see [Preemption flags on page 846](#)).

For example:

```

GUARANTEEDPREEMPTION TRUE
PREEMTPOLICY CHECKPOINT

QOSCFG[test1] QFLAGS=PREEMPTTEE MEMBERULIST=john PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=10000

```

## Related topics

- [Suspending jobs with preemption](#) on page 837
- [Requeueing jobs with preemption](#) on page 834
- [Canceling jobs with preemption](#) on page 830
- [Preemption flags](#) on page 846
- [About preemption](#) on page 828
- [PREEMTPOLICY types](#) on page 848
- [Testing and troubleshooting preemption](#) on page 852

## Requeueing jobs with preemption

### Context

*REQUEUE* is one of the [PREEMTPOLICY](#) types (for more information, see [PREEMTPOLICY types on page 848](#)). The *REQUEUE* value terminates active jobs and returns them to the job queue in an idle state.

For information about *PREEMPTTEE* and *PREEMPTOR* flags, see [Preemption flags on page 846](#)

The following outlines some benefits of using *REQUEUE* and also lists some things you should be aware of if you choose to use it.

#### Advantages:

- Jobs are automatically resubmitted into the job queue.

#### Cautions:

- A job gets resubmitted in the job queue at the same priority it had when Moab originally started it (i.e., the job does not jump ahead in the queue).
- Jobs start over from the beginning.



You must mark a job as *RESTARTABLE* if you want it to requeue. If you do not, the job will be canceled when it is preempted.

If supported by the resource manager, you can set the *RESTARTABLE* job flag when submitting the job by using the `msub -r` option. Otherwise, use the [JOBFLAGS](#) attribute of the associated class or QoS credential, as in this example:

```
CLASSCFG[low] JOBFLAGS=RESTARTABLE
```

For more information, see [Job Flags on page 145](#).

## To preempt jobs using REQUEUE

1. Make the following configurations to the `moab.cfg` file:
  - a. Set [GUARANTEEDPREEMPTION](#) to *TRUE*. (This causes Moab to lock *PREEMPTOR* jobs until [JOBRETRYTIME](#) expires.)

- b. Make sure that **JOBNODEMATCHPOLICY** is *not* set to **EXACTNODE**, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
- c. Set **PREEMTPOLICY** to **REQUEUE** (for more information, see [PREEMTPOLICY types on page 848](#)).
- d. Make sure that the **PREEMPTEE** job has a lower priority than the **PREEMPTOR** job (for more information, see [Preemption flags on page 846](#)).

For example:

```

GUARANTEEDPREEMPTION TRUE
PREEMTPOLICY REQUEUE

QOSCFG[test1] QFLAGS=PREEMPTEE JOBFLAGS=RESTARTABLE MEMBERULIST=john PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=10000
    
```

2. Submit a job to the preemptee QoS (test1). For example:

```
[john@g06]# echo sleep 600 | msub -l walltime=600 -l qos=test1 -l procs=128
```

(Optional) Examine the following output for showq:

```

Moab.1
[john@g06]# showq

active jobs-----
JOBID      USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.1     john       Running    128        00:09:59   Wed Nov 9 15:56:33

1 active job      128 of 128 processors in use by local jobs (100.00%)
                   2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUETIME

0 eligible jobs

blocked jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUETIME

0 blocked jobs

Total job: 1
    
```

3. Now submit a job to the preemptor QoS (test2). For example:

```
[john@g06]# echo sleep 600 | msub -l walltime=600 -l qos=test2 -l procs=128
```

(Optional) Examine the following output for showq and checkjob:

```

Moab.2
[john@g06]# showq

active jobs-----
JOBID      USERNAME      STATE      PROCS      REMAINING      STARTTIME
Moab.2     john          Running    128        00:09:59       Wed Nov 9 15:56:47

1 active job 128 of 128 processors in use by local jobs (100.00%)
  2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME      STATE      PROCS      WCLIMIT        QUEUE TIME
Moab.1     john          Idle       128        00:10:00       Wed Nov 9 15:56:33

1 eligible job

blocked jobs-----
JOBID      USERNAME      STATE      PROCS      WCLIMIT        QUEUE TIME

0 blocked jobs

Total jobs: 2

```

```

[john@g06]# checkjob Moab.2
job Moab.2

State: Running
Creds: user:john group:john qos:test2
WallTime: 00:02:04 of 00:10:00
SubmitTime: Wed Nov 9 15:56:46
(Time Queued Total: 00:00:01 Eligible: 00:00:00)

StartTime: Wed Nov 9 15:56:47
Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses
NodeCount: 2

Allocated Nodes:
node[01-02]*64

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.ELoX5Q

StartCount: 1
Flags: HASPREEMPTED,PREEMPTOR,GLOBALQUEUE,PROCSPECIFIED
StartPriority: 10000
Reservation 'Moab.2' (-00:02:21 -> 00:07:39 Duration: 00:10:00)

```

## Related topics

- [Suspending jobs with preemption on page 837](#)
- [Checkpointing jobs with preemption on page 833](#)
- [Canceling jobs with preemption on page 830](#)
- [Preemption flags on page 846](#)
- [About preemption on page 828](#)

- [PREEMTPOLICY types on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

## Suspending jobs with preemption

### Context

**SUSPEND** is one of the **PREEMTPOLICY** types (for more information, see [PREEMTPOLICY types on page 848](#)). The **SUSPEND** attribute causes active jobs to stop executing, but to remain in memory on the allocated compute nodes.

For information about **PREEMPTTEE** and **PREEMPTOR** flags, see [Preemption flags on page 846](#)

The following outlines some benefits of using **SUSPEND** and also lists some things you should be aware of if you choose to use it.

#### Advantages:

- The job remains in memory on the allocated compute nodes.
- Using **SUSPEND** frees up processor resources.
- The job can restart where it left off before it was suspended.

#### Cautions:

- There is a possibility that having multiple suspended jobs on a compute node will crash the swap.
- Moab tracks only *requested* memory of active jobs (not *used* memory). The swap can crash if the job uses a lot of memory and Moab starts other jobs.
- Suspended jobs do not relinquish their licenses.



You must mark a job as **SUSPENDABLE** if you want it to suspend. If you do not, the job will be requeued or canceled when it is preempted.

If supported by the resource manager, you can set the job **SUSPENDABLE** flag when submitting the job by using the `msub -r` option. Otherwise, use the **JOBFLAGS** attribute of the associated class or QoS credential, as in this example:

```
CLASSCFG[low] JOBFLAGS=SUSPENDABLE
```

For more information, see [Job Flags on page 145](#).

### To preempt jobs using SUSPEND

When you use **SUSPEND**, you must increase your **JOBRETRYTIME**. By default, **JOBRETRYTIME** is set to 60 seconds, but when you use **SUSPEND**, it is recommended that you increase the time to 300 seconds (5 minutes).

1. Make the following configurations to the `moab.cfg` file:
  - a. Set **GUARANTEEDPREEMPTION** to **TRUE**. (This causes Moab to lock **PREEMPTOR** jobs until **JOBRETRYTIME** expires.)
  - b. Make sure that **JOBNODEMATCHPOLICY** is *not* set to **EXACTNODE**, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
  - c. Set **PREEMTPOLICY** to **SUSPEND** (for more information, see [PREEMTPOLICY types on page 848](#)).
  - d. For the **PREEMPTEE** job, set **JOBFLAGS=RESTARTABLE,SUSPENDABLE**.
  - e. Make sure that the **PREEMPTEE** job has a lower priority than the **PREEMPTOR** job (for more information, see [Preemption flags on page 846](#)).

For example:

```

GUARANTEEDPREEMPTION TRUE
PREEMTPOLICY SUSPEND

QOSCFG[test1] QFLAGS=PREEMPTEE JOBFLAGS=RESTARTABLE,SUSPENDABLE MEMBERULIST=john
PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=10000

```

2. Submit a job to the preemptee QoS (`test1`). For example:

```
[john@g06]$ echo sleep 120 | msub -l procs=128,walltime=120 -l qos=test1
```

(Optional) Examine the output for `showq`:

```

Moab.7
[john@g06]# showq

active jobs-----
JOBID      USERNAME    STATE      PROCS      REMAINING    STARTTIME
Moab.7     john        Running    128        00:01:59     Thu Nov 10 12:28:44

1 active job      128 of 128 processors in use by local jobs (100.00%)
                   2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME

0 eligible jobs

blocked jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME

0 blocked jobs

Total job: 1

```

3. Now submit a job to the preemptor QoS (`test2`). For example:

```
[john@g06]$ echo sleep 120 | msub -l procs=128,walltime=120 -l qos=test2
```

(Optional) Examine the output for showq:

```
Moab.8
[john@g06]# showq

active jobs-----
JOBID      USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.7     john       Suspended  128        00:01:56   Thu Nov 10 12:28:44
Moab.8     john       Running    128        00:02:00   Thu Nov 10 12:28:48

2 active jobs 128 of 128 processors in use by local jobs (100.00%)
      2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 eligible jobs

blocked jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
0 blocked jobs

Total jobs: 2
```

*Note that when a job is suspended, it stays in the output of showq. This is normal behavior for a suspended job. Moab should only suspend a job once.*

4. (Optional) Examine the checkjob outputs for these two jobs.

```
[john@g06]$ checkjob Moab.9
job Moab.9

State: Suspended
Creds: user:john group:john qos:test1
WallTime: 00:00:02 of 00:02:00
SubmitTime: Thu Nov 10 12:36:29
(Time Queued Total: 00:00:07 Eligible: 00:00:00)

Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses
NodeCount: 2

Allocated Nodes:
node[01-02]*64

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.UFe8sQ

StartCount: 1
Flags: RESTARTABLE, SUSPENDABLE, PREEMPTEE, GLOBALQUEUE, PROCSPECIFIED
Attr: PREEMPTEE
StartPriority: 100
job cannot be resumed: preemption required but job is conditional preemptor with no
targets
BLOCK MSG: non-idle state 'Running' (recorded at last scheduling iteration)
```

```
[john@g06]$ checkjob Moab.10
job Moab.10

State: Running
Creds: user:john group:john qos:test2
WallTime: 00:00:00 of 00:02:00
SubmitTime: Thu Nov 10 12:36:31
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Thu Nov 10 12:36:31
Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses

Allocated Nodes:
node[01-02]*64

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.CZavjU

StartCount: 1
Flags: HASPREEMPTED,PREEMPTOR,GLOBALQUEUE,PROCSPECIFIED
StartPriority: 10000
Reservation 'Moab.10' (-00:00:07 -> 00:01:53 Duration: 00:02:00)
```

**i** Occasionally, Moab will keep a job from restarting, holding it in a suspended state for a long period of time, if it thinks the job cannot restart. For example, if a job could write to I/O before it was suspended, and now it cannot, Moab would realize the job is unable to start and would leave it in a suspended state.

## Related topics

- [Checkpointing jobs with preemption on page 833](#)
- [Queueing jobs with preemption on page 834](#)
- [Canceling jobs with preemption on page 830](#)
- [Preemption flags on page 846](#)
- [About preemption on page 828](#)
- [PREEMTPOLICY types on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

## Using owner preemption

### Context

Owner preemption allows jobs submitted by a reservation owner to preempt jobs submitted by other users (for more information, see [Configuring and Managing Reservations on page 460](#)).

Owner preemption is enabled with the [OWNERPREEMPT](#) reservation flag.

For information about [PREEMPTEE](#) and [PREEMPTOR](#) flags, see [Preemption flags on page 846](#)

## To enable owner preemption

1. Make the following configurations to the `moab.cfg` file:

- a. Set **GUARANTEEDPREEMPTION** to *TRUE*. (This causes Moab to lock *PREEMPTOR* jobs until JOBRETRYTIME expires.)
- b. Make sure that **JOBNODEMATCHPOLICY** is *not* set to *EXACTNODE*, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
- c. Set the **PREEMTPOLICY** type (for more information, see [PREEMTPOLICY types on page 848](#)).
- d. Set the **OWNERPREEMPT** flag.

**i** Optionally, if you want the owner preemption to override any PREEMPTMINTIME settings for *PREEMPTEE* jobs, you can set the OWNERPREEMPTIGNOREMINTIME flag as well.

e. Specify an owner.

**i** If the non-owner job does not have a *RESTARTABLE* or *REQUEUEABLE* flag set, the job will cancel.

For example:

```

GUARANTEEDPREEMPTION TRUE
PREEMTPOLICY <policy>

SRCFG[myrez] FLAGS=OWNERPREEMPT HOSTLIST=node01
SRCFG[myrez] OWNER=USER:john
SRCFG[myrez] USERLIST=jane,john PERIOD=INFINITY

QOSCFG[test1] QFLAGS=PREEMPTEE JOBFLAGS=restartable MEMBERULIST=john PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=10000

```

2. Submit a job to a user who is not the owner (in this example, jane).

```
[jane@g06]$ echo sleep 600 | msub -l walltime=600 -l procs=64
```

(Optional) Examine the following output for `showq` and `checkjob` for jane's job:

```

Moab.1
[jane@g06]$ showq

active jobs-----
JOBID      USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.1     jane      Running    64          00:09:57    Mon Nov 14 12:07:52

1 active job      64 of 64 processors in use by local jobs (100.00%)
                   1 of 1 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME

0 eligible jobs

blocked jobs-----
JOBID      USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME

0 blocked jobs

Total job: 1
    
```

```

root@g06]# checkjob Moab.1
job Moab.1

State: Running
Creds: user:jane group:jane
WallTime: 00:01:02 of 00:10:00
SubmitTime: Mon Nov 14 12:07:52
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Mon Nov 14 12:07:52
Total Requested Tasks: 64

Req[0] TaskCount: 64 Partition: FLEXlm
NodeCount: 1

Allocated Nodes:
[node01:64]

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.FoZfIU

StartCount: 1
Flags: GLOBALQUEUE,PROCSPECIFIED
StartPriority: 1
Reservation 'Moab.1' (-00:01:24 -> 00:08:36 Duration: 00:10:00)
    
```

3. Now submit a job for the owner (in this example, john).

```
[john@g06]$ echo sleep 600 | msub -l walltime=600 -l procs=50
```

```
[john@g06]$ echo sleep 600 | msub -l walltime=600 -l procs=50
```

(Optional) Examine the following output for showq and checkjob for john's job:

```

Moab.2
[john@g06]$ showq

active jobs-----
JOBID      USERNAME    STATE      PROCS      REMAINING    STARTTIME
Moab.1     jane        Canceling  64         00:07:43     Mon Nov 14 12:07:52
Moab.2     john        Running    50         00:09:59     Mon Nov 14 12:10:08

2 active jobs      64 of 64 processors in use by local jobs (100.00%)
1 of 1 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME

0 eligible jobs

blocked jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME

0 blocked jobs

Total jobs: 2
    
```

*Note that jane's job is canceled once john's job is submitted.*

```

[john@g06]$ checkjob Moab.2
job Moab.2

State: Running
Creds: user:john group:john
WallTime: 00:00:31 of 00:10:00
SubmitTime: Mon Nov 14 12:10:08
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Mon Nov 14 12:10:08
Total Requested Tasks: 50

Req[0] TaskCount: 50 Partition: FLEXlm
NodeCount: 1

Allocated Nodes:
[node01:50]

IWD: /opt/native
SubmitDir: /opt/native
Executable: /opt/native/spool/moab.job.jf1N4a

StartCount: 1
Flags: HASPREEMPTED, GLOBALQUEUE, PROCSPECIFIED
StartPriority: 1
Reservation 'Moab.2' (-00:00:48 -> 00:09:12 Duration: 00:10:00)
    
```

*Note the new **HASPREEMPTED** flag.*

(Optional) Now look at the showq for jane's job (after):

```
[root@g06]# checkjob Moab.1
job Moab.1

State: Removed
Completion Code: -1 Time: Mon Nov 14 12:10:08
Creds: user:jane group:jane
WallTime: 00:02:47 of 00:10:00
SubmitTime: Mon Nov 14 12:07:52
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

Total Requested Tasks: 64

Req[0] TaskCount: 64 Partition: FLEXlm
NodeCount: 1

Allocated Nodes:
[node01:64]

IWD: /opt/native
Executable: /opt/native/spool/moab.job.FoZfIU

Execution Partition: FLEXlm
Flags: GLOBALQUEUE,PROCSPECIFIED
StartPriority: 0
```

*Note that the state is now **Removed**.*

## Related topics

- [Preemption flags on page 846](#)
- [About preemption on page 828](#)
- [PREEMTPOLICY types on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

## Using QoS preemption

### Context

This section breaks down how to configure the `moab.cfg` file to set up preemption with QoS. Using QoS, you can specify preemption rules and control access to preemption privileges by using the `QFLAGS PREEMPT` and `PREEMPTOR` credentials. For information about the `PREEMPT` and `PREEMPTOR` flags, see [Preemption flags on page 846](#).

QoS-based preemption only occurs when the following three conditions are satisfied:

- The preemptor job has the `PREEMPTOR` attribute set.
- The preemptee job has the `PREEMPT` attribute set.
- The preemptor job has a higher priority than the preemptee job.

### To configure `moab.cfg` for QoS preemption

1. Set `GUARANTEEDPREEMPTION` to `TRUE`. (This causes Moab to lock `PREEMPTOR` jobs until `JOBRETRYTIME` expires.)

2. Make sure that [JOBNODEMATCHPOLICY](#) is *not* set to [EXACTNODE](#), which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
3. If it is not already, set [NODEACCESSPOLICY](#) to [SHARED](#).
4. Set the [PREEMPTPOLICY](#) policy type (for more information, see [PREEMPTPOLICY types on page 848](#)).
5. Set up [QFLAGS](#) to mark jobs as [PREEMPTTEE](#) (a lower-priority job that can be preempted by a higher-priority job), or as [PREEMPTOR](#) (a higher-priority job that can preempt a lower-priority job). As in the example:

```
QOSCFG[test1] QFLAGS=PREEMPTTEE MEMBERULIST=<user> PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=<user> PRIORITY=10000
```

For more information, see [Preemption flags on page 846](#).

6. Make sure that the [PREEMPTTEE](#) job has a lower priority than the [PREEMPTOR](#) job. As in the example:

```
QOSCFG[test1] QFLAGS=PREEMPTTEE MEMBERULIST=<user> PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=<user> PRIORITY=10000
```

For example:

```
GUARANTEEDPREEMPTION TRUE
PREEMPTPOLICY <policy>

QOSCFG[test1] QFLAGS=PREEMPTTEE MEMBERULIST=<user> PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=<user> PRIORITY=10000
```

## Related topics

- [About preemption on page 828](#)
- [Preemption Management on page 504](#)
- [Preemption flags on page 846](#)
- [PREEMPTPOLICY types on page 848](#)
- [Simple example of preemption on page 848](#)
- [Testing and troubleshooting preemption on page 852](#)

## References

### Manual preemption commands

You can use the [mjobctl](#) command to manually preempt jobs. The command can modify a job's execution state in the following ways:

Action	Flag	Details
Cancel	-c	Terminate job; remove from queue
Checkpoint	-C	Terminate and checkpoint job leaving job in queue
Requeue	-R	Terminate job; leave in queue
Resume	-r	Resume suspended job
Start (execute)	-x	Start idle job
Suspend	-s	Suspend active job

In general, users are allowed to suspend or terminate jobs they own. Administrators are allowed to suspend, terminate, resume, and execute any queued jobs.

Related topics

- [About preemption on page 828](#)
- [Testing and troubleshooting preemption on page 852](#)

## Preemption flags

Using QoS, you can specify preemption rules and control access to preemption privileges. This allows you to increase system throughput, improve job response time for specific classes of jobs, or enable various political policies. You enable all policies by specifying some QoS credentials with the [QFLAGS](#) *PREEMPT*, and others with *PREEMPTOR*.

PREEMPT	
<b>Description</b>	Indicates that the job can be preempted by a higher-priority job.
<b>Use</b>	Use for lower-priority jobs that can be preempted.
<b>Notes</b>	<div style="border: 1px solid #0056b3; border-radius: 10px; padding: 10px; background-color: #e6f2ff;">  This may delay some node actions. When reprovisioning, the system job may expire before the provision action occurs; while the action will still occur, the job will not show it.                 </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px;">QOSCFG[test1] QFLAGS=PREEMPT MEMBERULIST=&lt;user&gt; PRIORITY=100</pre>

PREEMPTOR	
<b>Description</b>	Indicates that the job should take priority and preempt any <i>PREEMPTEE</i> jobs.
<b>Use</b>	Use for jobs that need to take precedence over lower-priority jobs.
<b>Notes</b>	<p><b>i</b> <i>PREEMPTOR</i> jobs, either queued or running, must have a higher priority than <i>PREEMPTEE</i> jobs.</p> <p>When you configure job as a <i>PREEMPTOR</i>, you should also increase its priority (for details, see <a href="#">PREEMPTPRIOJOBSELECTWEIGHT</a> and <a href="#">PREEMPTRIMEWEIGHT</a>).</p>
<b>Example</b>	<code>QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=&lt;user&gt; PRIORITY=10000</code>

## Additional preemptor and preemptee information

**i** Preemptor priority plays a big role in preemption. Generally, you should assign the preemptor job a higher priority value than any other queued jobs so that it will move to (or near to) the top of the eligible queue.

You can set the [RESERVATIONPOLICY](#) parameter to *NEVER*. With this configuration, preemptee jobs can start whenever idle resources become available. These jobs will be allowed to run until a preemptor job arrives, at which point the preemptee jobs are preempted, freeing the resource. This configuration allows near immediate resource access for the preemptor jobs. Using this approach, a cluster can maintain near 100% system utilization while still delivering excellent turnaround time to the most important jobs.

In environments where job checkpointing or job suspension incur significant overhead, you might want to constrain the rate at which job preemption is allowed. You can use the [JOBPREEMPTMINACTIVETIME](#) parameter to throttle job preemption. In essence, this parameter prevents a newly started or newly resumed job from being eligible for preemption until it has executed for a specified amount of time. Conversely, you can exclude jobs from preemption after they have run for a certain amount of time by using the [JOBPREEMPTMAXACTIVETIME](#) parameter.

### Related topics

- [About preemption](#) on page 828
- [Using QoS preemption](#) on page 844
- [Manual preemption commands](#) on page 845
- [PREEMTPOLICY types](#) on page 848
- [Testing and troubleshooting preemption](#) on page 852

## PREEMTPOLICY types

You can use the [PREEMTPOLICY](#) parameter to control how the scheduler preempts a job. This parameter enforces preemption using one of the following methods:

PREEMTPOLICY type	Description
<b>SUSPEND</b>	Causes active jobs to stop executing, but to remain in memory on the allocated compute nodes.
<b>CHECKPOINT</b>	Saves the current job state and either terminates or continues running the job. A checkpointed job may restart at any time and resume execution from its most recent checkpoint.
<b>REQUEUE</b>	Terminates active jobs and returns them to the job queue in an idle state.
<b>CANCEL</b>	Cancels active jobs.

Each of these methods varies in the level of disruption to the job, *SUSPEND* being the least disruptive and *CANCEL* being the most disruptive.

Moab uses preemption escalation to free up resources. So for example, if the **PREEMTPOLICY** is set to *SUSPEND*, Moab uses this method if it is available; however, Moab will escalate it to something potentially more disruptive if necessary to preempt and free up resources.

### Related topics

- [Suspending jobs with preemption on page 837](#)
- [Checkpointing jobs with preemption on page 833](#)
- [Requeueing jobs with preemption on page 834](#)
- [Canceling jobs with preemption on page 830](#)
- [About preemption on page 828](#)
- [Preemption flags on page 846](#)

## Simple example of preemption

This section illustrates the process of setting up preemption on your system from beginning to end and contains examples of what actions to take and what you should see as you go.

### Example scenario

For this basic setup example, we will have a user who can submit to either a "test1" or "test2" QoS. This example will use a *QUEUE* preemption type.

We will go through three parts to set up this preemption:

- Configuring the `moab.cfg` file
- Submitting a job to the *PREEMPT* QoS
- Submitting a job to the *PREEMPTOR* QoS

Okay, let's get started!

## Configuring moab.cfg

First, you will need to make some configurations to the `moab.cfg` file.

1. Set **GUARANTEEDPREEMPTION** to *TRUE*. (This causes Moab to lock *PREEMPTOR* jobs until **JOBRETRYTIME** expires.)
2. Make sure that **JOBNODEMATCHPOLICY** is *not* set to *EXACTNODE*, which is not currently supported for preemption (for more information, see [Testing and troubleshooting preemption on page 852](#)).
3. Set the **PREEMTPOLICY** type. In this example, **PREEMTPOLICY** is set to *QUEUE*. For more information, see [PREEMTPOLICY types on page 848](#).
4. Set up **QFLAGS** to mark jobs as *PREEMPT* (a lower-priority job that can be preempted by a higher-priority job), or as *PREEMPTOR* (a higher-priority job that can preempt a lower-priority job). For more information, see [Preemption flags on page 846](#).

**i** For this example, we also set **JOBFLAGS=RESTARTABLE** (because this example uses *QUEUE*). For more information, see [Queueing jobs with preemption on page 834](#).

5. Make sure that the *PREEMPT* job has a lower priority than the *PREEMPTOR* job.

Here is an example of how that would all look in a `moab.cfg` file (text marked **red** for emphasis).

```
GUARANTEEDPREEMPTION TRUE
#should not be JOBNODEMATCHPOLICY EXACTNODE as it causes problems when starting jobs

PREEMTPOLICY QUEUE

QOSCFG[test1] QFLAGS=PREEMPT JOBFLAGS=RESTARTABLE MEMBERULIST=john PRIORITY=100
QOSCFG[test2] QFLAGS=PREEMPTOR MEMBERULIST=john PRIORITY=1000
```

Now you can submit a job to the preemptee QoS (*test1*).

## Submitting a job to the preemptee QoS

Let's submit a job to the preemptee QoS (*test1*), requesting all processor cores in the cluster:

```
[john@g06]# echo sleep 600 | msub -l walltime=600 -l qos=test1 -l procs=128
```

Take a look at the `showq` and `checkjob` output:

```
Moab.1
[john@g06]# showq

active jobs-----
JOBID      USERNAME    STATE      PROCS      REMAINING    STARTTIME
Moab.1     john        Running    128        00:09:59     Wed Nov 9 15:56:33

1 active job      128 of 128 processors in use by local jobs (100.00%)
2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME
0 eligible jobs

blocked jobs-----
JOBID      USERNAME    STATE      PROCS      WCLIMIT      QUEUE TIME
0 blocked jobs

Total job: 1
```

```
[john@g06]# checkjob Moab.1
job Moab.1

State: Running
Creds: user:john group:john qos:test1
WallTime: 00:00:00 of 00:10:00
SubmitTime: Wed Nov 9 15:56:33
(Time Queued Total: 00:00:00 Eligible: 00:00:00)

StartTime: Wed Nov 9 15:56:33
Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses

Allocated Nodes:
node[01-02]*64

IWD: /opt/native/
SubmitDir: /opt/native/
Executable: /opt/native/spool/moab.job.zOyf1N

StartCount: 1
Flags: RESTARTABLE,PREEMPTEE,GLOBALQUEUE,PROCSPECIFIED
Attr: PREEMPTEE
StartPriority: 100
Reservation 'Moab.1' (-00:00:03 -> 00:09:57 Duration: 00:10:00)
```

## Submitting a job to the preemptor QoS

Now we will submit a preemptor QoS job (*test2*) to preempt the first job (*test1*):

```
[john@g06]# echo sleep 600 | msub -l walltime=600 -l qos=test2 -l procs=128
```

Examine the following output for `showq` and `checkjob`:

```

Moab.2
[john@g06]# showq

active jobs-----
JOBID   USERNAME   STATE      PROCS      REMAINING   STARTTIME
Moab.2   john       Running    128        00:09:59   Wed Nov 9 15:56:47

1 active job 128 of 128 processors in use by local jobs (100.00%)
  2 of 2 nodes active (100.00%)

eligible jobs-----
JOBID   USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME
Moab.1   john       Idle       128        00:10:00   Wed Nov 9 15:56:33

1 eligible job

blocked jobs-----
JOBID   USERNAME   STATE      PROCS      WCLIMIT     QUEUE TIME

0 blocked jobs

Total jobs: 2

```

*Note that the preemptor job (Moab.2) moved to **Running**, while the preemptee job (Moab.1) was requested.*

```

[john@g06]# checkjob Moab.2
job Moab.2

State: Running
Creds: user:john group:john qos:test2
WallTime: 00:02:04 of 00:10:00
SubmitTime: Wed Nov 9 15:56:46
(Time Queued Total: 00:00:01 Eligible: 00:00:00)

StartTime: Wed Nov 9 15:56:47
Total Requested Tasks: 128

Req[0] TaskCount: 128 Partition: licenses
NodeCount: 2

Allocated Nodes:
node[01-02]*64

IWD: /opt/native/
SubmitDir: /opt/native/
Executable: /opt/native/spool/moab.job.ELoX5Q

StartCount: 1
Flags: HASPREEMPTED, PREEMPTOR, GLOBALQUEUE, PROCSPECIFIED
StartPriority: 10000
Reservation 'Moab.2' (-00:02:21 -> 00:07:39 Duration: 00:10:00)

```

*Note the flag, **HASPREEMPTED**. **HASPREEMPTED** is set when the **PREEMPTOR** job has preempted the **PREEMPTEE** job. Also note that the preemptor job priority plays a very big role in preemption. Generally, you should assign the preemptor a higher priority than any other queued jobs so that it will move to (or near to) the top of the eligible queue.*

## Related topics

- [About preemption on page 828](#)
- [Preemption flags on page 846](#)

- [PREEMTPOLICY types](#) on page 848
- [Manual preemption commands](#) on page 845
- [Testing and troubleshooting preemption](#) on page 852

## Testing and troubleshooting preemption

There are multiple steps associated with setting up a working preemption policy. With preemption, issues arise because it appears that Moab is not allowing preemptor jobs to preempt preemptee jobs in the right way. To diagnose this, use the following checklist:

Verify that preemptor jobs are marked with the <i>PREEMPTOR</i> flag. (Verify with <a href="#">checkjob &lt;JOBID&gt;   grep Flags</a> .)
Verify that preemptee jobs are marked with the <i>PREEMPTEE</i> flag. (Verify with <a href="#">checkjob &lt;JOBID&gt;   grep Flags</a> .)
Verify that the start priority of the preemptor job is higher than the priority of the preemptee job. (Verify with <a href="#">checkjob &lt;JOBID&gt;   grep Priority</a> .)
Verify that the resources allocated to the preemptee job match those requested by the preemptor job.
Verify that the preemptor job is within the 32-preemptee limit.
Verify that there are no policies preventing preemption from occurring. (Verify with <a href="#">checkjob -v -n &lt;NODEID&gt; &lt;JOBID&gt;</a> .)
Verify that the <a href="#">PREEMTPOLICY</a> parameter is properly set. (See <a href="#">PREEMTPOLICY types</a> on page 848.)
Verify that the preemptee job is properly marked as restartable, suspendable, or checkpointable. (Verify with <a href="#">checkjob &lt;JOBID&gt;   grep Flags</a> .)
Verify that <a href="#">GUARANTEEDPREEMPTION</a> is set to <i>TRUE</i> .
Verify that <a href="#">JOBNODEMATCHPOLICY</a> is <i>not</i> set to <i>EXACTNODE</i> . Moab does not currently consider <i>EXACTNODE</i> when it handles preemption, resulting in unexpected behavior when <i>EXACTNODE</i> is set in an environment with preemption.
Verify that <a href="#">NODEACCESSPOLICY</a> is <i>not</i> set to <i>SINGLEUSER</i> . ( <i>SHARED</i> is recommended.)
Verify that <a href="#">BACKFILLPOLICY</a> is set to <i>FIRSTFIT</i> .
Verify that the resource manager is properly responding to preemption requests. (Use <a href="#">mdiag -R</a> .)

If there is a resource manager level race condition, verify that Moab is properly holding target resources. (Verify with [mdiag -S](#) and set [RESERVATIONRETRYTIME](#) if needed.)

### Related topics

- [About preemption on page 828](#)
- [Quality of Service \(QoS\) Facilities on page 498](#)
- [Managing QoS Access on page 505](#)
- [JOBMAXPREEMPTPERITERATION on page 979](#)
- [Trigger components on page 765](#)
- [Checkpoint/Restart Facilities on page 528](#)
- [ENABLEFSVIOLATIONPREEMPTION on page 950](#)
- [PREEMPTPRIOJOBSELECTWEIGHT on page 1018](#)
- [PREEMPTSEARCHDEPTH on page 1019](#)
- [USAGEEXECUTIONTIMEWEIGHT on page 1064](#) (control priority of suspended jobs)
- [IGNOREPREEMPTEEPRIORITY on page 971](#) (relative job priority is ignored in preemption decisions)
- [DISABLESAMECREDPREEMPTION on page 945](#) (jobs cannot preempt other jobs with the same credential)
- [PREEMPTRTIMEWEIGHT on page 1018](#) (add remaining time of jobs to preemption calculation)

## Job templates

### About job templates

A Moab job template is a set of pre-configured settings, attributes, and resources that Moab applies to jobs that match certain criteria or to which you manually apply it. They perform three primary functions:

1. They generically match and categorize jobs.
2. They set arbitrary default or forced attributes for certain jobs.
3. They generate workflows that create and maintain user-requested services in a cloud environment.

You can use job templates in many aspects of scheduling [Peer-Based Grid](#) usage policies. Job templates are defined using the [JOB\\_CFG on page 975](#) configuration parameter.

Two methods exist for applying job templates to jobs. You can use the [JOBMATCHCFG on page 978](#) parameter to mark a template that contains the criteria a job must meet for eligibility and another template as the one to be applied to the job if it is eligible. This allows you to automate the use of templates. For example, to force all interactive jobs to run on a certain set of nodes, you can set one template (the criteria template) to have the *interactive* flag, then give the other template the desired host list. You can also apply a template directly to a job at submission if that ability is enabled for that template.

## Job template how-to's

- [Creating job templates](#) on page 854
- [Viewing job templates](#) on page 855
- [Applying templates based on job attributes](#) on page 855
- [Requesting job templates directly](#) on page 856
- [Creating workflows with job templates](#) on page 857

## Job template references

- [Job template extension attributes](#) on page 858
- [Job template matching attributes](#) on page 870
- [Job template examples](#) on page 870
- [Job template workflow examples](#) on page 871

## How-to's

### Creating job templates

#### Context

Job templates are created in the Moab configure file using the [JOB\\_CFG](#) on page 975 parameter.

#### To create a job template

1. Open `moab.cfg`. Add the **JOB\_CFG** parameter and give the new job template a unique name.

```
JOB_CFG[newtemplate]
```

2. Configure any desired attributes (see [Job template extension attributes](#) on page 858.). Some of the important attributes include:

- [FLAGS](#) on page 860 - Lets you specify any job flags that should be applied.

```
JOB_CFG[newtemplate] FLAGS=SUSPENDABLE
```

*When Moab applies `newtemplate` to a job, the job is marked as suspendable.*

- [SELECT](#) on page 866 - Lets you apply the template directly at job submission.

```
JOB_CFG[newtemplate] FLAGS=SUSPENDABLE SELECT=TRUE
```

*When you submit a job via `msub`, you can specify that your job has `newtemplate` applied to it. When Moab applies the template to a job, that job is marked as suspendable.*

- [TEMPLATEDPEND](#) on page 868 - Lets you create dependencies when you create a job template workflow (see [Creating workflows with job templates](#) on page 857.).

```
JOBCFG[newtemplate] FLAGS=SUSPENDABLE SELECT=TRUE TEMPLATEDEPEND=AFTER:job1.pre
```

*When Moab applies `newtemplate` to a job, the job cannot run until job `job1.pre` has finished running; the job is also marked as suspendable. You can specify that Moab apply this template to a job as you submit it.*

3. If you want to automate job template application, see [Applying templates based on job attributes on page 855](#) for instructions. If you want to apply the template manually on job submission, see [Requesting job templates directly on page 856](#) for instructions.

#### Related topics

- [Job template extension attributes on page 858](#)
- [Job template examples on page 870](#)

## Viewing job templates

### Context

You can view a job template by running the `mdiag -j` command.

### To view a job template

- Run the `mdiag -j` command with the `policy` flag. Moab returns a list of job templates configured in `moab.cfg`.

```
> mdiag -j --flags=policy --blocking
```

## Applying templates based on job attributes

### Context

The [JOBMATCHCFG on page 978](#) parameter allows you to establish relationships between a number of job templates. [JMAX](#) and [JMIN](#) function as filters to determine whether a job is eligible for a subsequent template to be applied to the job. If a job is eligible, [JDEF](#) and [JSET](#) templates apply attributes to the job. See [Job template extension attributes on page 858](#) for more information about the [JOBMATCHCFG](#) attributes. The table on that page indicates which job template types are compatible with which job template extension attributes.

**i** [JSETs](#) and [JDEFs](#) have only been tested using `msub` as the job submission command.

### To apply a job template based on job attributes

1. In the Moab configuration file, create a job template with a set of criteria that a job must meet in order for Moab to apply the template. In the following example, Moab will apply a template to all interactive jobs, so the first template sets the `interactive` flag.

```
JOBCFG[inter.min] FLAGS=interactive
```

2. Create the job template that Moab should apply to the job if it meets the requirements set in the first template. In this example, Moab ignores all configured policies, so the second template sets the *ignpolicies* flag.

```
JOBCFG[inter.set] FLAGS=ignpolicies
```

3. Use the **JOBMATCHCFG** parameter and its **JMAX** or **JMIN** (specify the template specifying maximum or minimum requirements) and **JDEF** or **JSET** (specify the template to be applied) attributes to demonstrate the relationship between the two templates (See [Job template matching attributes on page 870](#) for more information.). In this case, all interactive jobs ignore policies; in other words, if a submitted job has at least the *inter.min* template settings, Moab applies the *inter.set* template settings to the job.

```
JOBMATCHCFG[interactive] JMIN=inter.min JSET=inter.set
```

*Moab applies the inter.set template to all jobs with the interactive flag set, causing them to ignore Moab's configured policies.*

4. To control which job template is applied to a job that matches multiple templates, use **FLAGS=BREAK**. Job templates are processed in the order they are listed in the configuration file and using the **BREAK** flag causes Moab to stop evaluating **JOBMATCHCFG** entries that occur after the current match.

```
JOBMATCHCFG[small] JMIN=small.min JMAX=small.max JSET.set=small.set FLAGS=BREAK
JOBMATCHCFG[large] JMIN=large.min JMAX=large.max JSET=large.set
```

*In this case, the large template would not be applied when a job matches both the small and large templates. The small template matches first, and because of **FLAGS=BREAK**, Moab stops evaluating further **JOBMATCHCFG** entries for the job.*

## Related topics

- [Requesting job templates directly on page 856](#)
- [Job template examples on page 870](#)

## Requesting job templates directly

### Context

When a job template has its [SELECT on page 866](#) attribute set to **TRUE**, you can request that template directly on job submission.

### To directly request job templates

1. Set the **SELECT** attribute on the template in `moab.cfg`.

```
JOBCFG[medium.set] NODESET=ONEOF:FEATURE:fast,slow SELECT=true
```

2. Submit a job with a resource list (`msub -l`), requesting the template using the format `template=<templateName>`.

```
> msub -l template=medium.set
```

Moab creates a job with the *medium.set* job template created in step 1.

**i** Attributes set in the template are evaluated as if they were part of the job submission. They are still subject to all of the same ACLs and policies.

### Related topics

- [Applying templates based on job attributes on page 855](#)

## Creating workflows with job templates

### Context

Moab can create workflows from individual jobs using job templates.

### To build a workflow with job templates

1. Create the jobs in the workflow using the [JOBCFG on page 975](#) parameter (See [Creating job templates on page 854](#) for more information.). It might be useful to add the [PURGEONSUCCESSONLY on page 149](#) flag to your setup or destroy jobs; it will allow you to restart the jobs easily if they fail. Specify the order in which they should run with the [TEMPLATEDEPEND on page 868](#) attribute. Please see the [Job dependency syntax table](#) for a list of valid dependency options.

```
JOBCFG[setup.pre]   TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/setup.pre.sh
JOBCFG[setup.pre2]  TEMPLATEDEPEND=AFTER:setup.pre SELECT=TRUE
EXEC=/nfs/tools/setup.pre2.sh
JOBCFG[engineering] TEMPLATEDEPEND=AFTER:setup.pre2
```

*When Moab applies the engineering template to a qualifying job, the job will not run until template job setup.pre and then setup.pre2 are created from the specified EXEC strings and finish running.*

**i** The Moab naming convention for jobs created with job templates is `<moabId>.<templateName>`. By default, when Moab submits jobs to only one resource manager, the job IDs are synchronized with the resource manager's job IDs. You can use the parameter [USEMOABJOBID on page 1067](#) so that a template-created job is easily associated with its parent job (such as `moab.1.moab.1.setup.pre`).

2. Create the job template that will act as the criteria a job must meet for Moab to apply the *engineering* template. In this situation, the job must be submitted with the account name *engineering*.

```
JOBCFG[engineering.match] ACCOUNT=engineering
```

3. Create the [JOBMATCHCFG on page 978](#) configuration to tell Moab that when a job matches the *engineering.match* template, it should apply the *engineering* template.

```
JOBMATCHCFG[engineering.job] JMIN=engineering.match JSET=engineering
```

### Related topics

- [Job template extension attributes on page 858](#)
- [Job template workflow examples on page 871](#)
- [Creating job templates on page 854](#)

# References

## Job template extension attributes

When creating a job template, you can use any attribute acceptable within the [WIKI](#) workload query data format. In addition, job templates can use any of the extension attributes in the following table. Note that the Template type (**JMIN**, **JMAX**, **JDEF**, **JSET**) row indicates compatibility with the associated attribute (See [Applying templates based on job attributes on page 855](#) for more information.).

**i** Attributes set in a template are evaluated as if they were part of the original job submission. Their jobs are still subject to all the same ACLs and policies.

ACCOUNT	
<b>Format</b>	<ACCOUNT>[,<ACCOUNT>]...
<b>Template type</b>	<b>JMIN</b> <b>JDEF</b> <b>JSET</b>
<b>Description</b>	Account credentials associated with job. This is used for job template matching.
<b>Example</b>	<pre> JOBCFG[public] FLAGS=preemptee JOBCFG[public.min] ACCOUNT=public_acct JOBMATCHCFG[public] JMIN=public.min JSET=public                     </pre>

CLASS	
<b>Format</b>	<CLASS>[,<CLASS>]...
<b>Template type</b>	<b>JMIN</b> <b>JDEF</b> <b>JSET</b>
<b>Description</b>	Class credentials associated with job. This is used for job template matching.
<b>Example</b>	<pre> JOBCFG[night] FLAGS=preemptor JOBCFG[night.min] CLASS=night_class JOBMATCHCFG[night] JMIN=night.min JSET=night                     </pre>

CPUCLOCK	
<b>Format</b>	<STRING>
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JSET</b>
<b>Description</b>	CPU clock frequency for all CPUs of a job. For more information, see <a href="#">CPUCLOCK on page 621</a> . The job template extension overrides the job script.
<b>Example</b>	<pre> JOBCFG[slow] SELECT=TRUE cpuclock=1400 JOBCFG[fast] SELECT=TRUE cpuclock=3200  JOBCFG[cpu.min] CPUCLOCK=1000 JOBCFG[cpu.max] CPUCLOCK=2000 JOBCFG[cpu.set] CPUCLOCK=1500  JOBMATCHCFG[cpu] JMIN=cpu.min JMAX=cpu.max JSET=cpu.set </pre>

CPULIMIT	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JDEF</b> <b>JSET</b>
<b>Description</b>	Maximum amount of CPU time used by all processes in the job.
<b>Example</b>	<pre> JOBCFG[job.min] CPULIMIT=1:00:00:00 JOBCFG[job.max] CPULIMIT=2:00:00:00 </pre>

DESCRIPTION	
<b>Format</b>	<STRING>
<b>Template type</b>	<b>JMAX</b> <b>JDEF</b>
<b>Description</b>	Description of the job. When you run the <a href="#">checkjob</a> command, the description appears as Reason.
<b>Example</b>	<pre> JOBCFG[webdb] DESCRIPTION="Template job" </pre>

DPROCS	
<b>Format</b>	<INTEGER>
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JSET</b>
<b>Description</b>	Number of processors dedicated per task. The default is 1.
<b>Example</b>	<pre>JOBCFG [job.min] DPROCS=2 JOBCFG [job.max] DPROCS=4</pre>

EXEC	
<b>Format</b>	<STRING>
<b>Template type</b>	<b>JSET</b>
<b>Description</b>	Specifies what the job runs, regardless of what the user set.
<b>Example</b>	<pre>JOBCFG [setup.pre] EXEC=nfs/tools/setup.pre.sh</pre>

FLAGS	
<b>Format</b>	<JOBFLAG>[,<JOBFLAG>]...
<b>Template type</b>	<b>JMIN</b> <b>JDEF</b> <b>JSET</b>
<b>Description</b>	One or more legal <a href="#">job flag</a> values.
<b>Example</b>	<pre>JOBCFG [webdb] FLAGS=NORMSTART</pre>

GNAME	
<b>Format</b>	<STRING>

GNAME	
<b>Template type</b>	JDEF JSET
<b>Description</b>	Group credential associated with job.
<b>Example</b>	<pre>JOBCFG[webserv] GNAME=service</pre> <p> For matching the group, see the <a href="#">GROUP</a> attribute.</p>

GRES	
<b>Format</b>	<genericResource>[:<COUNT>][,<genericResource>[:<COUNT>]]...
<b>Template type</b>	JMAX JDEF
<b>Description</b>	Consumable generic attributes associated with individual nodes or the special pseudo-node <a href="#">global</a> , which provides shared cluster (floating) consumable resources. Use the <a href="#">NODECFG</a> parameter to configure such resources.
<b>Example</b>	<pre>JOBCFG[gres.set] GRES=abacus:2</pre> <p><i>In this example, the gres.set template applies two Abaqus licenses per task to a matched job.</i></p>

GROUP	
<b>Format</b>	<GROUP>[,<GROUP>]...
<b>Template type</b>	JMIN
<b>Description</b>	Group credentials associated with job. This is used for job template matching.
<b>Example</b>	<pre>JOBCFG[webserv] GROUP=service</pre> <p> For information about setting the group, see the <a href="#">GNAME</a> attribute.</p>

MEM	
<b>Format</b>	<INTEGER>
<b>Template type</b>	JMIN JMAX JDEF JSET
<b>Description</b>	Maximum amount of physical memory per task used by the job in megabytes. You can optionally specify other units with your integer (300kb or 2gb, for example). See <a href="#">Requesting resources on page 2586</a> for more information.
<b>Example</b>	<pre>JOBCFG[smalljobs] MEM=25</pre>

NODEACCESSPOLICY	
<b>Format</b>	One of the following: <a href="#">SHARED</a> , <a href="#">SHAREDONLY</a> , <a href="#">SINGLEJOB</a> , <a href="#">SINGLETASK</a> , <a href="#">SINGLEUSER</a> , or <a href="#">UNIQUEUSER</a>
<b>Template type</b>	JDEF JSET
<b>Description</b>	Specifies how node resources will be shared by a job. See the <a href="#">Node Access Policies on page 403</a> for more information.
<b>Example</b>	<pre>JOBCFG[serverapp] NODEACCESSPOLICY=SINGLEJOB</pre>

NODERANGE	
<b>Format</b>	<MIN>[,<MAX>]
<b>Template type</b>	JMAX JDEF
<b>Description</b>	Minimum and maximum nodes allowed to be allocated to job.
<b>Example</b>	<pre>JOBCFG[vizserver] NODERANGE=1,16</pre>

NODES	
<b>Format</b>	<INTEGER>
<b>Template type</b>	JMIN JMAX JSET
<b>Description</b>	Number of nodes required by the job. The default is 1. See <a href="#">Node Definition</a> for more information.
<b>Example</b>	<pre>JOBCFG[job.min] NODES=2 JOBCFG[job.max] NODES=4</pre>

NODESET	
<b>Format</b>	<STRING>
<b>Template type</b>	JSET
<b>Description</b>	See <a href="#">Node Set Overview on page 514</a> for more information.
<b>Example</b>	<pre>JOBCFG[medium.set] NODESET=ONEOF:FEATURE:fast,slow</pre>

PARTITION	
<b>Format</b>	<PARTITION>[:<PARTITION>]...
<b>Template type</b>	JMIN JDEF JSET
<b>Description</b>	Specifies the partition (or partitions) in which a job must run.
<b>Example</b>	<pre>JOBCFG[meis] PARTITION=math:geology</pre>

PREF	
<b>Format</b>	<FEATURE>[,<FEATURE>]...
<b>Template type</b>	JDEF JSET
<b>Description</b>	Specifies which node features are preferred by the job and should be allocated if available. See <a href="#">PREF</a> for more information.
<b>Example</b>	<code>JOBCFG [meis] PREF=bigmem</code>

PRIORITY	
<b>Format</b>	<INTEGER>
<b>Template type</b>	JMAX JDEF
<b>Description</b>	Relative job priority.  <div style="border: 1px solid #0056b3; padding: 5px; background-color: #e6f2ff;">  <b>PRIORITY</b> works only as a default setting and not as an override (<b>JSET</b>) setting.         </div>
<b>Example</b>	<code>JOBCFG [meis] PRIORITY=25000</code>

PROCRANGE	
<b>Format</b>	<MIN>[,<MAX>]
<b>Template type</b>	JDEF JSET
<b>Description</b>	Minimum and maximum processors allowed to be allocated to job.
<b>Example</b>	<code>JOBCFG [meis] PROCRANGE=2, 64</code>

QOS	
<b>Format</b>	<QOS>[,<QOS>]...
<b>Template type</b>	JMIN JDEF JSET
<b>Description</b>	QoS credentials associated with job. This is used for job template matching.
<b>Example</b>	<pre>JOBCFG[admin] RFEATURES=bigmem JOBCFG[admin.min] QOS=admin_qos JOBMATCHCFG[admin] JMIN=admin.min JSET=admin</pre>

RARCH	
<b>Format</b>	<STRING>
<b>Template type</b>	JSET
<b>Description</b>	Architecture required by job.
<b>Example</b>	<pre>JOBCFG[servapp] RARCH=i386</pre>

RFEATURES	
<b>Format</b>	<FEATURE>[,<FEATURE>]...
<b>Template type</b>	JMIN JDEF JSET
<b>Description</b>	List of features required by job.
<b>Example</b>	<pre>JOBCFG[servapp] RFEATURES=fast, bigmem</pre>

RM	
<b>Format</b>	<STRING>
<b>Template type</b>	JDEF JSET
<b>Description</b>	Destination resource manager to be associated with job.
<b>Example</b>	<code>JOBCFG[webdb] RM=slurm</code>

ROPSYS	
<b>Format</b>	<STRING>
<b>Template type</b>	JDEF JSET
<b>Description</b>	Operating system required by job.
<b>Example</b>	<code>JOBCFG[test.set] ROPSYS=windows</code>

SELECT	
<b>Format</b>	<BOOLEAN> : TRUE   FALSE
<b>Description</b>	Job template can be directly requested by job at submission.
<b>Example</b>	<code>JOBCFG[servapp] SELECT=TRUE</code>

SOFTWARE	
<b>Format</b>	<RESTYPE>[+ :}<COUNT>][@<TIMEFRAME>]
<b>Template type</b>	JDEF JSET

SOFTWARE	
<b>Description</b>	Indicates generic resources required by the job. See <a href="#">SOFTWARE</a> for more information.
<b>Example</b>	<pre>JOBCFG[servapp] SOFTWARE=matlab:2</pre>

SYSTEMJOBTYPE	
<b>Template type</b>	<b>JMIN</b>
<b>Description</b>	System job type (ex. vmcreate).
<b>Example</b>	<pre>JOBCFG[vmcreate.min] SYSTEMJOBTYPE=vmcreate JOBCFG[vmcreate.set] TRIGGER=atype=reserve,action="00:05:00",etype=end JOBMATCHCFG[vmcreate] JMIN=vmcreate.min JSET=vmcreate.set</pre>

TASKS	
<b>Format</b>	<INTEGER>
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JSET</b>
<b>Description</b>	Number of tasks required by job. The default is 1. See <a href="#">Task Definition</a> for more information.
<b>Example</b>	<pre>JOBCFG[job.min] TASKS=4 JOBCFG[job.max] TASKS=8</pre>

TASKPERNODE	
<b>Format</b>	<INTEGER>
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JDEF</b>

TASKPERNODE	
<b>Description</b>	Exact number of tasks required per node. The default is 0. <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; margin-top: 10px;">  <b>TASKPERNODE</b> works only as a default setting and not as an override (<b>JSET</b>) setting.                 </div>
<b>Example</b>	<pre> JOB_CFG[job.min] TASKPERNODE=2 JOB_CFG[job.max] TASKPERNODE=4                     </pre>

TEMPLATEDEPEND	
<b>Format</b>	<TYPE>:<TEMPLATE_NAME>
<b>Description</b>	Create another job from the <TEMPLATE_NAME> job template, on which any jobs using this template will depend. This is used for dynamically creating workflows. See <a href="#">Job Dependencies</a> for more information.
<b>Example</b>	<pre> JOB_CFG[engineering] TEMPLATEDEPEND=AFTER:setup.pre JOB_CFG[setup.pre] SELECT=TRUE EXEC=/tools/setup.pre.sh                     </pre>

UNAME	
<b>Format</b>	<STRING>
<b>Default</b>	<b>JDEF</b> <b>JSET</b>
<b>Description</b>	User credential associated with job.
<b>Example</b>	<pre> JOB_CFG[webserv] UNAME=service                     </pre> <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; margin-top: 10px;">  For matching the user, see the <a href="#">USER</a> attribute.                 </div>

USER	
<b>Format</b>	<USER>[,<USER>]...

USER	
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b>
<b>Description</b>	User credentials associated with job.
<b>Example</b>	<pre>JOBCFG[webserv] USER=service</pre> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 5px;">  For setting the user, see the <a href="#">UNAME</a> attribute.         </div>

VARIABLE	
<b>Format</b>	<NAME>[:<VAL>]
<b>Template type</b>	<b>JMIN</b> <b>JSET</b>
<b>Description</b>	Variables attached to the job template.
<b>Example</b>	<pre>JOBCFG[this] VARIABLE=var1:1 VARIABLE=var2:1</pre> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 5px;">  Variables are set upon successful completion of the job.         </div>

WCLIMIT	
<b>Format</b>	[[HH:]MM:]SS
<b>Template type</b>	<b>JMIN</b> <b>JMAX</b> <b>JDEF</b> <b>JSET</b>
<b>Description</b>	Walltime required by job. The default is 8640000 (100 days).
<b>Example</b>	<pre>JOBCFG[job.min] WCLIMIT=2:00:00 JOBCFG[job.max] WCLIMIT=12:00:00</pre>

Related topics

- [Job template examples on page 870](#)
- [Creating job templates on page 854](#)

## Job template matching attributes

The **JOBMATCHCFG** on page 978 parameter allows you to establish relationships between a number of job templates. The table in [Job template extension attributes on page 858](#) indicates which job template types are compatible with which job template extension attributes. The following types of templates can be specified with the **JOBMATCHCFG** parameter:

Attribute	Description
<b>JMAX</b>	<p>A potential job is rejected if it has matching attributes set or has resource requests that exceed those specified in this template.</p> <div style="border: 1px solid #005596; padding: 5px; margin-top: 10px;"> <p> For <b>JMAX</b>, a job template can specify only positive non-zero numbers as maximum limits for generic resources. If a job requests a generic resource that is not limited by the template, then the template can still be used.</p> </div>
<b>JMIN</b>	A potential job is rejected if it does not have matching attributes set or has resource requests that do not meet or exceed those specified in this template.
<b>JDEF</b>	A matching job has the specified attributes set as defaults but all values can be overridden by the user if the matching attribute is explicitly set at job submission time.
<b>JSET</b>	A matching job has the specified attributes forced to these values and these values override any values specified by the submitter at job submission time.
<b>JSTAT</b>	A matching job has its usage statistics reported into this template.

Related topics

- [Job template extension attributes on page 858](#)
- [Job template examples on page 870](#)
- [Applying templates based on job attributes on page 855](#)

## Job template examples

Job templates can be used for a wide range of purposes including enabling automated learning, setting up custom application environments, imposing special account constraints, and applying group default settings. The following examples highlight some of these uses:

*Example 3-164: Setting up application-specific environments*

```
JOBCFG[xxx] EXEC=*app* JOBPROLOG=/usr/local/appprolog.x
```

**Example 3-165: Applying job preferences and defaults**

```

JOBCFG[xxx] CLASS=appq EXEC=*app* PREF=clearspeed
NODEALLOCATIONPOLICY PRIORITY
NODECFG[DEFAULT] PRIORITYF=5.0*PREF

```

**Example 3-166: Applying resource constraints to fuzzy collections**

In the following example, a job template match is set up. Using the [JOBMATCHCFG on page 978](#) parameter, Moab is configured to apply all attributes of the *inter.set* job template to all jobs that match the constraints of the *inter.min* job template. In this example, all interactive jobs are assigned the *ignpolicies* flag that allows them to ignore active, idle, system, and partition level policies. Interactive jobs are also locked into the test standing reservation and thus only allowed to run on the associated nodes.

```

# limit all users to a total of two non-interactive jobs
USERCFG[DEFAULT] MAXJOB=2
SRCFG[test] DESCRIPTION="compute pool for interactive and short duration jobs"
SRCFG[test] JOBATTRLIST=INTERACTIVE
SRCFG[test] MAXTIME=1:00:00
SRCFG[test] HOSTLIST=R:atl[16-63]
JOBCFG[inter.min] FLAGS=interactive
JOBCFG[inter.set] FLAGS=ignpolicies
JOBMATCHCFG[interactive] JMIN=inter.min JSET=inter.set

```

**Example 3-167: Resource manager templates**

In the following example, interactive jobs are not allowed to enter through this resource manager and any job that does route in from this resource manager interface has the *preemptee* flag set.

```

JOBCFG[no_inter] FLAGS=interactive
JOBCFG[preempt_job] FLAGS=preemptee
RMCFG[gridA.in] MAX.JOB=no_inter SET.JOB=preempt_job

```

**Related topics**

- [Job template extension attributes on page 858](#)
- [Job template workflow examples on page 871](#)
- [Creating job templates on page 854](#)

**Job template workflow examples****Example 3-168: A workflow with multiple dependencies**

In this example the job will depend on the completion of two other jobs Moab creates. Both jobs execute at the same time.

```

# Engineering2
JOBCFG[engineering2] TEMPLATEDEPEND=AFTER:engineering2.pre2
TEMPLATEDEPEND=AFTER:engineering2.pre
JOBCFG[engineering2.pre2] TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/engineering2.pre2.sh
JOBCFG[engineering2.pre] TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/engineering2.pre.sh
JOBCFG[engineering2.match] ACCOUNT=engineering2
JOBMATCHCFG[engineering2.job] JMIN=engineering2.match JSET=engineering2

```

*Example 3-169: Jobs that run after the submission job*

Three additional jobs are created that depend on the submitted job.

```
# Workflow 2
JOBCFG[workflow2] TEMPLATEDEPEND=BEFORE:workflow2.post1
TEMPLATEDEPEND=BEFORE:workflow2.post2 TEMPLATEDEPEND=BEFORE:workflow2.post3
JOBCFG[workflow2.post1] TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow2.post1.sh
JOBCFG[workflow2.post2] TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow2.post2.sh
JOBCFG[workflow2.post3] TASKS=2 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow2.post3.sh
JOBCFG[workflow2.match] ACCOUNT=workflow2
JOBMATCHCFG[workflow2.job] JMIN=workflow2.match JSET=workflow2
```

*Example 3-170: A complex workflow*

A complex workflow that handles failures.

```
# Workflow 4
JOBCFG[workflow4.step1] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step1.sh TEMPLATEDEPEND=BEFOREFAIL:workflow4.fail1
JOBCFG[workflow4.fail1] TASKS=1 WCLIMIT=00:00:30 SELECT=TRUE
EXEC=/usr/tools/workflow.fail.1.sh TEMPLATEDEPEND=BEFOREANY:workflow4.fail2
JOBCFG[workflow4.fail2] TASKS=1 WCLIMIT=00:00:30 SELECT=TRUE
EXEC=/usr/tools/workflow.fail.2.sh
# Submission job
JOBCFG[workflow4.step2] TEMPLATEDEPEND=AFTEROK:workflow4.step1
TEMPLATEDEPEND=BEFOREOK:workflow4.step3.1 TEMPLATEDEPEND=BEFOREOK:workflow4.step3.2
JOBCFG[workflow4.step3.1] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step3.1.sh
JOBCFG[workflow4.step3.2] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step3.2.sh TEMPLATEDEPEND=BEFOREOK:workflow4.step4
JOBCFG[workflow4.step4] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step4.sh
JOBCFG[workflow4.step4] TEMPLATEDEPEND=BEFOREOK:workflow4.step5.1
TEMPLATEDEPEND=BEFOREOK:workflow4.step5.2 TEMPLATEDEPEND=BEFORENOTOK:workflow4.step5.3
JOBCFG[workflow4.step5.1] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step5.1.sh
JOBCFG[workflow4.step5.2] TASKS=1 WCLIMIT=00:01:00 SELECT=TRUE
EXEC=/usr/tools/workflow.step5.2.sh
JOBCFG[workflow4.step5.3] TASKS=1 WCLIMIT=00:00:30 SELECT=TRUE
EXEC=/usr/tools/workflow.step5.3.sh
JOBCFG[workflow4.match] ACCOUNT=workflow4
```

## Related topics

- [Creating workflows with job templates on page 857](#)
- [Applying templates based on job attributes on page 855](#)
- [Job template examples on page 870](#)
- [Job template extension attributes on page 858](#)

## Moab Workload Manager for Grids



[Cluster Consolidation and Sovereign Grids](#) is a video tutorial of a session offered at Moab Con that offers further details for understanding cluster consolidation and sovereign grids.

Moab Grid Scheduler allows sites to establish relationships among multiple clusters. There are three types of relationships you can implement within the grid: (1) centralized management, (2) hierarchal management, and (3) localized management. These relationships provide access to additional resources, improve load-balancing, provide single system images, and offer other benefits. The grid interface is flexible allowing sites to establish the needed relationship.

- [Grid Basics on page 873](#)
- [Grid Configuration Basics on page 881](#)
- [Centralized Grid Management \(Master/Slave\) on page 882](#)
- [Hierarchal Grid Management on page 882](#)
- [Localized Grid Management on page 884](#)
- [Resource Control and Access on page 885](#)
- [Workload Submission and Control on page 888](#)
- [Reservations in the Grid on page 888](#)
- [Grid Usage Policies on page 889](#)
- [Grid Scheduling Policies on page 891](#)
- [Grid Credential Management on page 893](#)
- [Grid Data Management on page 895](#)
- [Grid Security on page 900](#)
- [Grid Diagnostics and Validation on page 900](#)

## Grid Basics

- [Grid Overview](#)
- [Grid Benefits](#)
- [Scalability](#)
- [Resource Access](#)
- [Load-Balancing](#)
- [Single System Image \(SSI\)](#)
- [High Availability](#)
- [Grid Relationships](#)
  - [Grid Relationships](#)
  - [Hierarchal Management](#)
  - [Local Management](#)

- [Submitting Jobs to the Grid](#)
- [Viewing Jobs and Resources](#)

## Grid Overview

A grid enables you to exchange workload and resource status information and to distribute jobs and data among clusters in an established relationship. In addition, you can use resource reservations to mask reported resources, coordinate requests for consumable resources, and quality of service guarantees.

In a grid, some servers running Moab are a source for jobs (that is, where users, portals, and other systems submit jobs), while other servers running Moab are a destination for these jobs (that is, where the jobs execute). Thus, jobs originate from a source server and move to a destination server. For a source server to make an intelligent decision, though, resource availability information must flow from a destination server to that source server.

Because you can manage workload on both the source and destination side of a grid relationship, you have a high degree of control over exactly when, how, and where to execute workload.

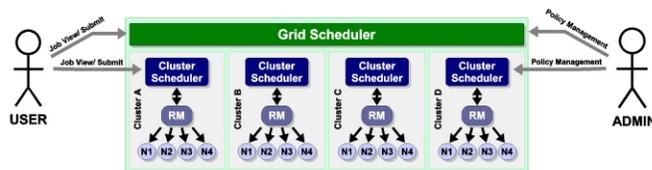
## Grid Benefits

Moab's peer-to-peer capabilities can be used for multiple purposes, including any of the following:

- manage access to external shared resources
- enable cluster monitoring information services
- enable massive-scalability clusters
- enable distributed grid computing

Of these, the most common use is the creation of grids to join multiple centrally managed, partially autonomous, or fully autonomous clusters. The purpose of this section is to highlight the most common uses of grid technology and provide references to sections which further detail their configuration and management. Other sections cover the standard aspects of grid creation including configuring [peer relationships](#), enabling [data staging](#), [credential management](#), [usage policies](#), and other factors.

Image 3-11: Jobs submitted to grid scheduler then cluster schedulers



## Management-Scalability

Much like a massive-scalability cluster, a massive-scalability grid allows organizations to overcome scalability limitations in resource managers, networks, message passing libraries, security middleware, file systems, and other forms of software and hardware infrastructure. Moab does this by allowing a single large set of resources to be broken into multiple smaller, more manageable clusters, and then

virtually re-assembling them using Moab. Moab becomes responsible for integrating the seams between the clusters and presenting a single-system image back to the end-users, administrators, and managers.

 Jobs cannot span clusters.

## Resource Access

In some cases, the primary motivation for creating a grid is to aggregate resources of different types into a single system. This aggregation allows for multi-step jobs to run a portion of the job on one architecture, and a portion on another.

A common example of a multi-architecture parameter-sweep job would be a batch regression test suite which requires a portion of the tests running on Redhat 7.2, a portion on SUSE 9.1, a portion on Myrinet nodes, and a portion on Infiniband nodes. While it would be very difficult to create and manage a single cluster which simultaneously provided all of these configurations, Moab can be used to create and manage a single grid which spans multiple clusters as needed.

## Load-Balancing

While grids often have additional motivations, it is rare to have a grid created where increased total system utilization is not an objective. By aggregating the total pool of jobs requesting resources and increasing the pool of resources available to each job, Moab is able to improve overall system utilization, sometimes significantly. The biggest difficulty in managing multiple clusters is preventing inter-cluster policies and the cost of migration from overwhelming the benefits of decreased fragmentation losses. Even though remote resources may be available for immediate usage, migration costs can occur in the form of credential, job, or data staging and impose a noticeable loss in responsiveness on grid workload.

Moab provides tools to allow these costs to be monitored and managed and both cluster and grid level performance to be reported.

## Single System Image (SSI)

Another common benefit of grids is the simplicity associated with a single system image-based resource pool. This simplicity generally increases productivity for end-users, administrators, and managers.

An SSI environment tends to increase the efficiency of end-users by minimizing human errors associated with porting a request from a known system to a less known system. Additionally, the single point of access grid reduces human overhead associated with monitoring and managing workload within multiple independent systems.

For system administrators, a single system image can reduce overhead, training time, and diagnostic time associated with managing a cluster. Furthermore, with Moab's peer-to-peer technology, no additional software layer is required to enable the grid and no new tools must be learned. No additional layers means no additional failure points, and that is good for everyone involved.

Managers benefit from SSI by being able to pursue organization mission objectives globally in a more coordinated and unified manner. They are also able to monitor progress toward those objectives and effectiveness of resources in general.

## High Availability

A final benefit of grids is their ability to decrease the impact of failures. Grids add another layer of high availability to the cluster-level high availability. For some organizations, this benefit is a primary motivation, pulling together additional resources to allow workload to continue to be processed even in the event that some nodes, or even an entire cluster, become unavailable. Whether the resource unavailability is based on node failures, network failures, systems middleware, systems maintenance, or other factors, a properly configured grid can reroute priority workload throughout the grid to execute on other compatible resources.

With grids, there are a number of important factors in high availability that should be considered:

- enabling highly available job submission/job management interfaces
- avoiding network failures with redundant routes to compute resources
- handling partial failures
- dynamically restarting failed jobs

## Grid Relationships

There are three types of relationships you can implement within the grid:

- [Centralized Management \(Master/Slave\)](#)
- [Centralized/Localized Management \(Hierarchal\)](#)
- [Localized Management \(Peer-to-Peer\)](#)

### Centralized Management (Master/Slave)

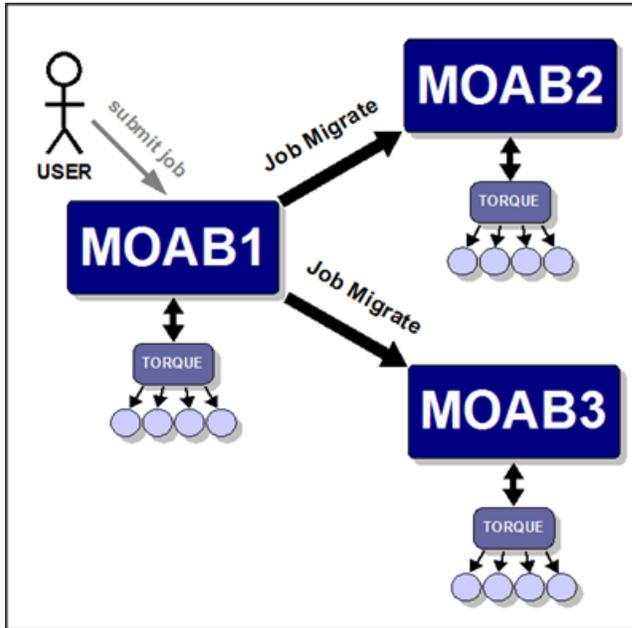
The centralized management model (master/slave) allows users to submit jobs to a centralized source server running Moab. The source Moab server obtains full resource information from all clusters and makes intelligent scheduling decisions across all clusters. Jobs (and [data](#) when configured to do so) are distributed to the remote clusters as needed. The centralized management model is recommended for intra-organization grid environments when cluster autonomy is not as necessary.

In the centralized management (master-slave) configuration, roles are clear. In other configurations, individual Moab servers may simultaneously act as sources to some clusters and destinations to others or as both a source and a destination to another cluster.

#### Example of the Centralized Management (Master/Slave) Model

XYZ Research has three clusters - MOAB1, MOAB2, and MOAB3--running Moab and the [TORQUE](#) resource manager. They would like to submit jobs at a single location (cluster MOAB1) and have the jobs run on whichever cluster can provide the best responsiveness.

The desired behavior is essentially a *master-slave* relationship. MOAB1 is the central, or master, cluster. On MOAB1, resource managers point to the local TORQUE resource manager and to the Moab servers on cluster MOAB2 and cluster MOAB3. The Moab servers on MOAB2 and MOAB3 are configured to trust cluster MOAB1 and to execute in slave mode.



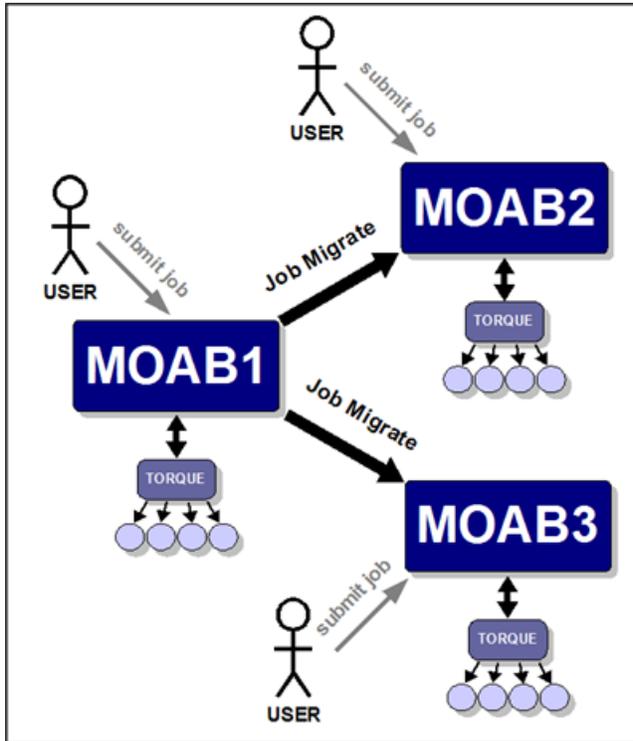
With this configuration, XYZ Research may submit jobs to the master Moab server running on cluster MOAB1 and may, as stated earlier, submit jobs from the slave nodes as well. However, only the master Moab server may schedule jobs. For example, cluster MOAB2 and cluster MOAB3 cannot schedule a job, but they can accept a job and retain it in an idle state until the master directs it to run.

**i** You can turn off job submission on slave nodes by setting the [DISABLESLAVEJOBSUBMIT](#) on page 945 parameter to *TRUE*.

The master Moab server obtains full resource information from all three clusters and makes intelligent scheduling decisions and distributes jobs (and [data](#) when configured to do so) to the remote clusters. The Moab servers running on clusters MOAB2 and MOAB3 are destinations behaving like a local resource manager. The Moab server running on MOAB1 is a source, loading and using this resource information.

## Centralized/Localized Management (Hierarchal)

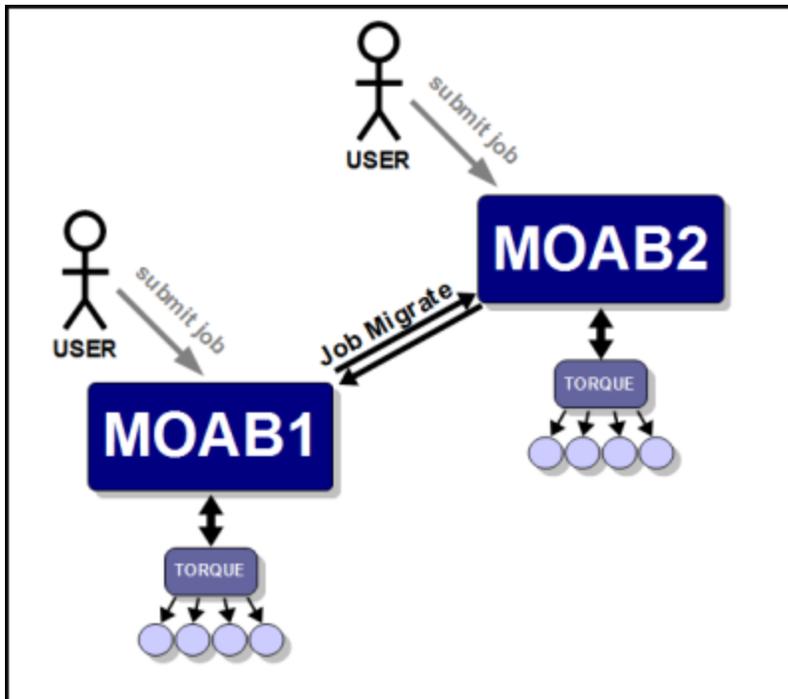
As with the centralized management model (master/slave), the hierarchal model allows users to submit jobs to a centralized source server running Moab. However, in the hierarchal model, clusters retain sovereignty, allowing local job scheduling. Thus, if communication between the source and destination clusters is interrupted, the destination cluster(s) can still run jobs locally.



In the hierarchal model, the source Moab server obtains full resource information from all clusters and makes intelligent scheduling decisions across all clusters. As needed, jobs and data are distributed to the remote clusters. Or, if preferred, a destination cluster may also serve as its own source; however, a destination cluster may not serve as a source to another destination cluster. The centralized management model is recommended for intra-organization grid environments when cluster autonomy and/or local management is necessary.

### Localized Management (Peer-to-Peer)

The localized management (peer-to-peer) model allows you to submit jobs on one cluster and schedule the jobs on the other cluster (it currently works with two clusters). For example, a job may be submitted on MOAB1 and run on MOAB2. Jobs can also migrate in the opposite direction (that is, from MOAB2 to MOAB1). The source servers running Moab obtain full resource information from both clusters and make intelligent scheduling decisions across both clusters. Jobs (and [data](#) when configured to do so) are migrated to other clusters as needed.



**i** Jobs will not migrate indefinitely. The localized management model limits them to one migration.

This model allows clusters to retain their autonomy while still allowing jobs to run on either cluster. No central location for job submission is needed, and you do not need to submit jobs from different nodes based on resource needs. You can submit a job from any location and it is either migrated to nodes on the least utilized cluster or the cluster requested in the job submission. This model is recommended for grids in an inter-organization grid environment.

## Submitting Jobs to the Grid

In any peer-to-peer or grid environment where jobs must be migrated between clusters, use the Moab `msub` command. Once a job has been submitted to Moab using `msub`, Moab identifies potential destinations and migrates the job to the destination cluster.

Using Moab's `msub` job submission command, jobs may be submitted using PBS or LSF command file syntax and be run on any cluster using any of the resource managers. For example, a PBS job script may be submitted using `msub` and depending on availability, Moab may translate a subset of the job's directives and execute it on an LSF cluster.

**i** Moab can only stage/migrate jobs between resource managers (in between clusters) that have been submitted using the `msub` command. If jobs are submitted directly to a low-level resource manager, such as PBS, Moab will still be able to schedule them but only on resources directly managed by the resource manager to which they were submitted.

### Example 1

A small pharmaceutical company, BioGen, runs two clusters in a centralized relationship. The slave is an older IBM cluster running Loadleveler, while the master manages the slave and also directly manages a large Linux cluster running TORQUE. A new user familiar with LSF has multiple LSF job scripts he would like to continue using. To enable this, the administrators make a symbolic link between the Moab `msub` client and the file `bsub`. The user begins submitting his jobs via `bsub` and, according to availability, the jobs run on either the Loadleveler or TORQUE clusters.

**Example 2**

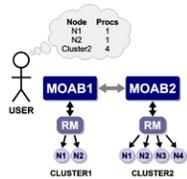
A research lab wants to use spare cycles on its four clusters, each of which is running a local resource manager. In addition to providing better site-wide load balancing, the goal is to also provide some of its users with single point access to all compute resources. Various researchers have made it clear that this new multi-cluster load balancing must not impose any changes on users who are currently using these clusters by submitting jobs locally to each cluster.

In this example, the scheduler mode of the destination clusters should be set to *NORMAL* rather than *SLAVE*. In *SLAVE* mode, Moab makes no local decisions - it simply follows the directions of remote trusted peers. In *NORMAL* mode, each Moab is fully autonomous, scheduling all local workload and coordinating with remote peers when and how to schedule migrated jobs.

From the perspective of a local cluster user, no new behaviors are seen. Remote jobs are migrated in from time to time, but to the user each job looks as if it were locally submitted. The user continues to submit, view, and manage jobs as before, using existing local jobs scripts.

## Viewing Jobs and Resources

By default, each destination Moab server will report all compute nodes it finds back to the source Moab server. These reported nodes appear within the source Moab as local nodes each within a partition associated with the resource manager reporting them. If a source resource manager was named `slave1`, all nodes reported by it would be associated with the `slave1partition`. Users and administrators communicating with the source Moab via [Moab Cluster Manager](#), [Moab Access Portal](#), or standard Moab command line tools would be able to view and analyze all reported nodes.



**i** The grid view will be displayed if either the source or the destination server is configured with grid view.

For job information, the default behavior is to only report to the source Moab information regarding jobs that originated at the source. If information about other jobs is desired, this can be configured as shown in the [Workload Submission and Control](#) section.

Related topics

- [Resource Control and Access](#)

## Grid Configuration Basics

- [Peer Configuration Overview](#)
- [Initial Configuration](#)
- [Viewing Jobs From Other Peers](#)

### Peer Configuration Overview

In the simplest case, establishing a peer relationship can be accomplished with as few as two configuration lines: one line to indicate how to contact the peer and one line to indicate how to authenticate the server. However, data migration issues, credential mapping, and usage policies must often be addressed in order to make a peer-based grid effective.

To address these issues Moab provides facilities to control how peers inter-operate, enabling full autonomy over both client and server ends of the peer relationship.

### Initial Configuration

At a minimum, only two parameters must be specified to establish a peer relationship: [RMCFG on page 1035](#) and [CLIENTCFG\[<X>\] on page 937](#). **RMCFG** allows a site to specify interface information directing Moab on how to contact and inter-operate with the peer. For peer interfaces, a few guidelines must be followed with the **RMCFG** parameter:

- the **TYPE** attribute of the peer must be set to *moab*
- the **SERVER** attribute must point to the host and user interface port of the remote Moab server
- the *name* of the resource manager should match the name of the remote peer cluster as specified with the [SCHEDCFG on page 1039](#) parameter in the peer `moab.cfg`.

```
# moab.cfg on MoabServer01
SCHEDCFG[MoabServer01] MODE=NORMAL SERVER=hpc-01:41111
RMCFG[MoabServer02]     TYPE=moab   SERVER=hpc-02:40559
...
```

Configuring the **CLIENTCFG** parameter is mandatory. When specifying the **CLIENTCFG** parameter for peers, the following guidelines must be followed:

- the **CLIENTCFG** parameter must be specified in the `moab-private.cfg` file on both peers
- an **RM:** prefix is required before the peer's name
- if using default secret key based security, the value of the **KEY** attribute must match the **KEY** value set on the corresponding remote peer
- the **AUTH** attribute must be set to *admin1* in the `moab-private.cfg` on the destination Moab

```
# moab-private.cfg on MoabServer01
CLIENTCFG[RM:MoabServer02] KEY=3esfv0=32re2-tdbne
....
```

```
# moab-private.cfg on MoabServer02
CLIENTCFG[RM:MoabServer01] KEY=3esfv0=32re2-tdbne AUTH=admin1
...
```

## Centralized Grid Management (Master/Slave)

### Master Configuration

The process of setting up the master configuration is the same as setting up a [source Moab configuration](#). The master/slave relationship is configured in each `moab.cfg` on the slave.

```
# moab.cfg on Master
SCHEDCFG[master] SERVER=master:42559 MODE=NORMAL
...
```

```
# moab-private.cfg on Master
CLIENTCFG[RM:slave1] KEY=3esfv0=32re2-tdbne
...
```

### Slave Configuration

The slave's relationship with the master is determined by the **MODE**. Setting **MODE** to **SLAVE** notifies the master to take control of starting jobs on the slave. The master starts the jobs on the slave. In **SLAVE** mode, jobs can be submitted locally to the slave, but are not seen or started by the master. When a job is submitted locally to the slave the job is locked into the cluster and cannot migrate to other clusters.

```
# moab.cfg on Slave
SCHEDCFG[slave1] SERVER=slave1:42559 MODE=SLAVE
...
```

```
# moab-private.cfg on Slave
CLIENTCFG[RM:master] KEY=3esfv0=32re2-tdbne AUTH=admin1
...
```

## Hierarchical Grid Management

- [Configuring a Peer Server \(Source\)](#)
  - [Simple Hierarchical Grid](#)

## Configuring a Peer Server (Source)

Peer relationships are enabled by creating and configuring a [resource manager](#) interface using the [RMCFG](#) parameter. This interface defines how a given Moab will load resource and workload information and enforce its scheduling decisions. In non-peer cases, the [RMCFG](#) parameter points to a resource manager such as TORQUE, LSF, or SGE. However, if the [TYPE](#) attribute is set to *moab*, the [RMCFG](#) parameter can be used to configure and manage a peer relationship.

## Simple Hierarchal Grid

The first step to create a new peer relationship is to configure an interface to a destination Moab server. In the following example, cluster C1 is configured to be able to *see* and *use* resources from two other clusters.

```
SCHEDCFG [C1]  MODE=NORMAL  SERVER=head.C1.xyz.com:41111
RMCFG [C2]    TYPE=moab    SERVER=head.C2.xyz.com:40559
RMCFG [C3]    TYPE=moab    SERVER=head.C3.xyz.com:40559
...
```

*C1 allows a global view of the underlying clusters. From C1, jobs can be viewed and modified. C2 and C3 act as separate scheduling entities that can receive jobs from C1. C1 migrates jobs to C2 and C3 based on available resources and policies of C1. Jobs migrated to C2 and C3 are scheduled according to the policies on C2 and C3.*

In this case, one [RMCFG](#) parameter is all that is required to configure each peer relationship if standard secret key based authentication is being used and a shared default secret key exists between the source and destination Moabs. However, if peer relationships with multiple clusters are to be established and a per-peer secret key is to be used (highly recommended), then a [CLIENTCFG](#) parameter must be specified for the authentication mechanism. Because the secret key must be kept secure, it must be specified in the `moab-private.cfg` file. For the current example, a per-peer secret key could be set up by creating the following `moab-private.cfg` file on the C1 cluster.

```
CLIENTCFG [RM:C2]  KEY=fastclu3t3r
CLIENTCFG [RM:C3]  KEY=14436aaa
```

**i** The key specified can be any alphanumeric value and can be locally generated or made up. The only critical aspect is that the keys specified on each end of the peer relationship match.

Additional information can be found in the [Grid Security](#) section which provides detailed information on designing, configuring, and troubleshooting peer security.

Continuing with the example, the initial source side configuration is now complete. On the destination clusters, C2 and C3, the first step is to configure authentication. If a shared default secret key exists between all three clusters, then configuration is complete and the clusters are ready to communicate. If per-peer secret keys are used (recommended), then it will be necessary to create matching `moab-private.cfg` files on each of the destination clusters. With this example, the following files would be required on C2 and C3 respectively:

```
CLIENTCFG [RM:C1]  KEY=fastclu3t3r  AUTH=admin1
```

```
CLIENTCFG [RM:C1]  KEY=14436aaa  AUTH=admin1
```

Once peer security is established, a final optional step would be to configure scheduling behavior on the destination clusters. By default, each destination cluster accepts jobs from each trusted peer. However, it will also be fully autonomous, accepting and scheduling locally submitted jobs and enforcing its own local policies and optimizations. If this is the desired behavior, then configuration is complete.

In the current example, with no destination side scheduling configuration, jobs submitted to cluster C1 can run locally, on cluster C2 or on cluster C3. However, the established configuration does not necessarily enforce a strict master-slave relationship because each destination cluster (C2 and C3) has complete autonomy over how, when, and where it schedules both local and remote jobs. Each cluster can potentially receive jobs that are locally submitted and can also receive jobs from other source Moab servers. See [Slave Mode](#) for more information on setting up a master-slave grid.

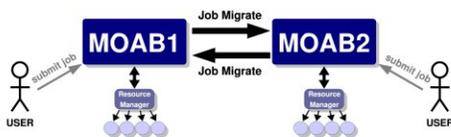
Further, each destination cluster will accept any and all jobs migrated to it from a trusted peer without limitations on who can run, when and where they can run, or how many resources they can use. If this behavior is either too restrictive or not restrictive enough, then destination side configuration will be required.

## Localized Grid Management

- [Enabling Bi-Directional Job Flow](#)
  - [True Peer-to-Peer Grid](#)

### Enabling Bi-Directional Job Flow

Image 3-12: Bi-directional peer-to-peer setup



For each peer interface, an [RMCFG on page 1035](#) parameter is only required for the server (or source side of the interface). If two peers are to share jobs in both directions, the relationship is considered to be bi-directional.

### True Peer-to-Peer Grid

Previous examples involved grid masters that coordinated the activities of the grid and made it so direct contact between peers was not required. However, if preferred, the master is not required and individual clusters can interface directly with each other in a true peer manner. This configuration is highlighted in the following example:

```
# Cluster A
SCHEDCFG[clusterA] MODE=NORMAL SERVER=clusterA
RMCFG[clusterA] TYPE=pbs
RMCFG[clusterB] TYPE=moab SERVER=clusterB:40559
CLIENTCFG[RM:clusterB] AUTH=admin1 KEY=banana16
```

```
# Cluster B
SCHEDCFG[clusterB] MODE=NORMAL SERVER=clusterB
RMCFG[clusterB] TYPE=pbs
RMCFG[clusterA] TYPE=moab SERVER=clusterA:40559
CLIENTCFG[RM:clusterA] AUTH=admin1 KEY=banana16
```

**i** If you are using Moab Accounting Manager, the *Start* action is not supported as a non-blocking accounting action in Peer-to-Peer grids. You will need to include *Start* as a blocking action. For example:

```
AMCFG[mam] BLOCKINGACTIONS=Start
```

## Resource Control and Access

- [Controlling Resource Information](#)
  - [Direct Node View](#)
  - [Mapped Node View](#)
  - [Managing Queue Visibility over the Grid](#)
- [Managing Resources with Grid Sandboxes](#)
  - [Controlling Access on a Per Cluster Basis](#)
  - [Access Control Lists/Granting Access to Local Jobs](#)

### Controlling Resource Information

In a Moab peer-to-peer grid, resources can be viewed in one of two models:

- [Direct](#) - nodes are reported to remote clusters exactly as they appear in the local cluster
- [Mapped](#) - nodes are reported as individual nodes, but node names are mapped to a unique name when imported into the remote cluster

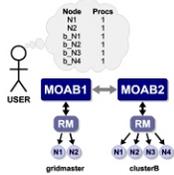
#### Direct Node View

Direct node import is the default resource information mode. No additional configuration is required to enable this mode.

#### Mapped Node View

In this mode, nodes are reported just as they appear locally by the exporting cluster. However, on the importing cluster side, Moab maps the specified node names using the resource manager [object map](#). In an object map, node mapping is specified using the `node` keyword as in the following example:

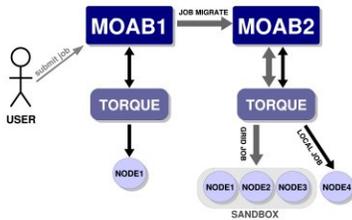
```
SCHEDCFG[gridmaster] MODE=NORMAL
RMCFG[clusterB] TYPE=moab OMAP=file://$HOME/clusterb.omap.dat
...
node:b_*,*
```



In this example, all nodes reported by *clusterB* have the string *b\_* prepended to prevent node name space conflicts with nodes from other clusters. For example, if cluster *clusterB* reported the nodes *node01*, *node02*, and *node03*, cluster *gridmaster* would report them as *b\_node01*, *b\_node02*, and *b\_node03*.

See [object mapping](#) for more information on creating an object map file.

## Managing Resources with Grid Sandboxes



A cluster may wish to participate in a grid but may desire to dedicate only a set amount of resources to external grid workload or may only want certain peers to have access to particular sets of resources. With Moab, this can be achieved by way of a grid sandbox which must be configured at the destination cluster. Grid sandboxes can both constrain external resource access and limit which resources are reported to other peers. This allows a cluster to only report a defined subset of its total resources to source peers and restricts peer workload to the sandbox. The sandbox can be set aside for peer use exclusively, or can allow local workload to also run inside of it. Through the use of multiple, possibly overlapping grid sandboxes, a site may fully control resource availability on a per peer basis.

A grid sandbox is created by configuring a [standing reservation](#) on a destination peer and then specifying the *ALLOWGRID* flag on that reservation. This flag tells the Moab destination peer to treat the standing reservation as a grid sandbox, and, by default, only the resources in the sandbox are visible to grid peers. Also, the sandbox only allows workload from other peers to run on the contained resources.

**Example 3-171: Dedicated Grid Sandbox**

```

SRCFG[sandbox1] PERIOD=INFINITY HOSTLIST=node01,node02,node03
SRCFG[sandbox1] CLUSTERLIST=ALL FLAGS=ALLOWGRID
...

```

*The standing reservation `sandbox1` creates a grid sandbox which always exists and contains the nodes `node01`, `node02`, and `node03`. This sandbox will only allow grid workload to run within it by default. This means that the scheduler will not consider the boxed resources for local workload.*

Grid sandboxes inherit all of the same power and flexibility that standing reservations have. See [Managing Reservations](#) for additional information.

**i** The flag `ALLOWGRID` marks the reservation as a grid sandbox and as such, it precludes grid jobs from running anywhere else. However, it does *not* enable access to the reserved resources. The `CLUSTERLIST` attribute in the above example enables access to all remote jobs.

## Controlling Access on a Per Cluster Basis

Often clusters may wish to control which peers are allowed to use certain sandboxes. For example, Cluster A may have a special contract with Cluster B and will let overflow workload from Cluster B run on 60% of its resources. A third peer in the grid, Cluster C, doesn't have the same contractual agreement, and is only allowed 10% of Cluster A at any given time. Thus two separate sandboxes must be made to accommodate the different policies.

```

SRCFG[sandbox1] PERIOD=INFINITY HOSTLIST=node01,node02,node03,node04,node05
SRCFG[sandbox1] FLAGS=ALLOWGRID CLUSTERLIST=ClusterB
SRCFG[sandbox2] PERIOD=INFINITY HOSTLIST=node06 FLAGS=ALLOWGRID
SRCFG[sandbox2] CLUSTERLIST=ClusterB,ClusterC,ClusterD USERLIST=ALL
...

```

*This example configuration illustrates how cluster A could set up their sandboxes to follow a more complicated policy. In this policy, `sandbox1` provides exclusive access to nodes 1 through 5 to jobs coming from peer ClusterB by including `CLUSTERLIST=ClusterB` in the definition. Reservation `sandbox2` provides shared access to `node6` to local jobs and to jobs from clusters B, C, and D through use of the `CLUSTERLIST` and `USERLIST` attributes.*

With this setup, the following policies are enforced:

- local jobs may see all nodes and run anywhere except nodes 1 through 5
- jobs from cluster B may see and run only on nodes 1 through 6
- jobs from clusters C and D may see and run only on node 6

As shown in the example above, sandboxes can be shared across multiple peers by listing all sharing peers in the `CLUSTERLIST` attribute (comma delimited).

## Access Control Lists/Granting Access to Local Jobs

It is not always desirable to have the grid sandbox reserve resources for grid consumption, exclusively. Many clusters may want to use the grid sandbox when local workload is high and demand from the grid is relatively low. Clusters may also wish to further restrict what kind of grid workload can run in a sandbox. This fine-grained control can be achieved by attaching access control lists (ACLs) to grid sandboxes.

Since sandboxes are basically special standing reservations, the syntax and rules for specifying an ACL is identical to those found in [Managing Reservations](#).

### Example

```
SRCFG[sandbox2] PERIOD=INFINITY HOSTLIST=node04,node05,node06
SRCFG[sandbox2] FLAGS=ALLOWGRID QOSLIST=high GROUPLIST=engineer
...
```

*A cluster decides to dedicate resources to a sandbox, but wishes local workload to also run within it. An additional ACL is then associated with the definition. The reservation `sandbox2` takes advantage of this feature by allowing local jobs running with a QOS of `high`, or under the group `engineer`, to also run on the sandboxed nodes `node04`, `node05`, and `node06`.*

## Workload Submission and Control

- [Controlling Peer Workload Information](#)
- [Determining Resource Availability](#)

### Controlling Peer Workload Information

By default, a peer is only responsible for workload that is submitted via that particular peer. This means that when a source peer communicates with destination peers it only receives information about workload it sent to those destination peers. If desired, the destination peers can send information about *all* of its workload: both jobs originating locally and remotely. This is called *local workload exporting*. This may help simplify administration of different clusters by centralizing monitoring and management of jobs at one peer.

To implement local workload exporting, use the `LOCALWORKLOADEXPORT` resource manager flag. For example:

```
RMCFG[ClusterA.INBOUND] FLAGS=LOCALWORKLOADEXPORT # source peer
...
```

*This example shows the configuration on a destination peer (ClusterB) that exports its local and remote workload to the source peer (ClusterA).*

**i** LOCALWORDKLOADEXPORT does not need to be configured in master/slave grids.

### Related topics

- [Job Start Time Estimates](#)

## Reservations in the Grid

In some environments, globally-shared resources may need to be managed to guarantee the full environment required by a particular job. Resources such as networks, storage systems, and license managers may be used only by batch workload but this workload may be distributed among multiple independent clusters. Consequently, the jobs from one cluster may utilize resources required by jobs

from another. Without a method of coordinating the needs of the various cluster schedulers, resource reservations will not be respected by other clusters and will be of only limited value.

Using the centralized model, Moab allows the importing and exporting of reservations from one peer server to another. With this capability, a source peer can be set up for the shared resource to act as a clearinghouse for other Moab cluster schedulers. This source peer Moab server reports configured and available resource state and in essence possesses a global view of resource reservations for all clusters for the associated resource.

To allow the destination peer to export reservation information to the source Moab, the [RMCFG on page 1035](#) lines for all client resource managers must include the flag *RSVEXPORT*. The source Moab should be configured with a resource manager interface to the destination peer and include both the *RSVEXPORT* and *RSVIMPORT* flags. For the destination peer, *RSVEXPORT* indicates that it should *push* information about newly created reservations to the source Moab, while the *RSVIMPORT* flag indicates that the source Moab server should import and locally enforce reservations detected on the destination peer server.

## Grid Usage Policies

- [Grid Usage Policy Overview](#)
- [Peer Job Resource Limits](#)
- [Usage Limits via Peer Credentials](#)
- [Using General Policies in a Grid Environment](#)
  - [Source Cluster Policies](#)

### Grid Usage Policy Overview

Moab allows extensive control over how peers interact. These controls allow the following:

- Limiting which remote users, group, and accounts can utilize local compute resources
- Limiting the total quantity of local resources made available to remote jobs at any given time
- Limiting remote resource access to a specific subset of resources
- Limiting timeframes during which local resources will be made available to remote jobs
- Limiting the types of remote jobs which will be allowed to execute

### Peer Job Resource Limits

Both source and destination peers can limit the types of jobs they will allow in terms of resources requested, services provided, job duration, applications used, etc using Moab's job template feature. Using this method, one or more job profiles can be created on either the source or destination side, and Moab can be configured to allow or reject jobs based on whether or not the jobs meet the specified job profiles.

When using the **ALLOWJOBLIST** and **REJECTJOBLIST** attributes, the following rules apply:

- All jobs that meet the job templates listed by **ALLOWJOBLIST** are allowed.
- All jobs that do not meet **ALLOWJOBLIST** job templates and which do meet **REJECTJOBLIST** job templates are rejected.
- All jobs that meet no job templates in either list are allowed.

## Usage Limits via Peer Credentials

With peer interfaces, destination clusters willing to accept remote jobs can [map](#) these jobs onto a select subset of users, accounts, QoSs, and queues. With the ability to lock these jobs into certain credentials comes the ability to apply any arbitrary credential constraints, priority adjustments, and resource limitations normally available within cluster management. Specifically, the following can be accomplished:

- limit number of active jobs simultaneously allowed
- limit quantity of allocated compute resources simultaneously allowed
- adjust job priority
- control access to specific scheduling features (deadlines, reservations, preemption, etc)
- adjust fairshare targets
- limit resource access

## Using General Policies in a Grid Environment

While Moab does provide a number of unique grid-based policies for use in a grid environment, the vast majority of available management tools come from the transparent application of cluster policies. Cluster-level policies such as [job prioritization](#), [node allocation](#), [fairshare](#), [usage limits](#), [reservations](#), [preemption](#), and allocation management all just work and can be applied in a grid in exactly the same manner.

The one key concept to understand is that in a centralized based grid, these policies apply across the entire grid; in a peer-based grid, these policies apply only to local workload and resources.

## Source Cluster Policies

In many cases, organizations are interested in treating jobs differently based on their point of origin. This can be accomplished by assigning and/or keying off of a unique credential associated with the remote workload. For example, a site may wish to constrain jobs from a remote cluster to only a portion of the total available cluster cycles. This could be accomplished using usage limits, fairshare targets, fairshare caps, reservations, or allocation management based policies.

The examples below show three different approaches for constraining remote resource access.

**Example 3-172: Constraining Remote Resource Access via Fairshare Caps**

```
# define peer relationship and map all incoming jobs to orion account
RMCFG[orion.INBOUND] SET.JOB=orion.set
JOBCFG[orion.set] ACCOUNT=orion
# configure basic fairshare for 7 one day intervals
FSPOLICY DEDICATEDPS
FSINTERVAL 24:00:00
FSDEPTH 7
FSUSERWEIGHT 100
# use fairshare cap to limit jobs from orion to 10% of cycles
ACCOUNTCFG[orion] FSCAP=10%
```

**Example 3-173: Constraining Remote Resource Access via Fairshare Targets and Preemption**

```
# define peer relationship and map all incoming jobs to orion account RMCFG
[orion.INBOUND] SET.JOB=orion.set
JOBCFG[orion.set] ACCOUNT=orion
# local cluster can preempt jobs from orion
USERCFG[DEFAULT] JOBFLAGS=PREEMPTOR
PREEMTPOLICY CANCEL
# configure basic fairshare for 7 one day intervals
FSPOLICY DEDICATEDPS
FSINTERVAL 24:00:00
FSDEPTH 7
FSUSERWEIGHT 100
# decrease priority of remote jobs and force jobs exceeding 10% usage to be
preemptible
ACCOUNTCFG[orion] FSTARGET=10-
ENABLEFSVIOLATIONPREEMPTION TRUE
```

**Example 3-174: Constraining Remote Resource Access via Priority and Usage Limits**

```
# define peer relationship and map all incoming jobs to orion account RMCFG
[orion.INBOUND] SET.JOB=orion.set
JOBCFG[orion.set] QOS=orion
USERCFG[DEFAULT] QDEF=orion
# local cluster can preempt jobs from orion
USERCFG[DEFAULT] JOBFLAGS=PREEMPTOR
PREEMTPOLICY CANCEL
# adjust remote jobs to have reduced priority
QOSCFG[orion] PRIORITY=-1000
# allow remote jobs to use up to 64 procs without being preemptible and up to 96 as
preemtees
QOSCFG[orion] MAXPROC=64,96
ENABLESPVIOLATIONPREEMPTION TRUE
```

**Related topics**

- [Grid Sandbox](#) - control grid resource access

## Grid Scheduling Policies

- [Peer-to-Peer Resource Affinity Overview](#)
- [Peer Allocation Policies](#)
- [Per-partition Scheduling](#)

## Peer-to-Peer Resource Affinity Overview

The concept of resource affinity stems from a number of facts:

- Certain compute architectures are able to execute certain compute jobs more effectively than others.
- From a given location, staging jobs to various clusters may require more expensive allocations, more data and network resources, and more use of system services.
- Certain compute resources are owned by external organizations and should be used sparingly.

Regardless of the reason, Moab servers allow the use of peer resource affinity to guide jobs to the clusters that make the best fit according to a number of criteria.

At a high level, this is accomplished by creating a number of job templates and associating the profiles with different peers with varying impacts on estimated execution time and peer affinity.

## Peer Allocation Policies

A direct way to assign a peer allocation algorithm is with the [PARALLOCATIONPOLICY](#) parameter. Legal values are listed in the following table:

Value	Description
<b>FirstStart</b>	Allocates resources from the eligible peer that can start the job the soonest.
<b>LoadBalance</b>	Allocates resources from the eligible peer with the most available resources; measured in tasks (balances workload distribution across potential peers).
<b>LoadBalanceP</b>	Allocates resources from the eligible peer with the most available resources; measured in percent of configured resources (balances workload distribution across potential peers).
<b>Random</b>	Allocates partitions in a random order each iteration. In general, all the jobs scheduled within the same iteration receive the same randomized list of partitions. This means the randomization happens between iterations and not within the same iteration. One iteration Moab might start with partition X and the next it might start with partition Y.
<b>RoundRobin</b>	Allocates resources from the eligible peer that has been least recently allocated.



The `mdiag -t -v` command can be used to view current calculated partition priority values.

## Per-partition Scheduling

Per-partition scheduling can be enabled by adding the following lines to `moab.cfg`:

```
PERPARTITIONSCHEDULING TRUE
JOBMIGRATEPOLICY JUSTINTIME
```

To use per-partition scheduling, you must configure fairshare trees where particular users have higher priorities on one partition, and other users have higher priorities on a different partition.

**i** Do not set the [USEANYPARTITIONPRIO](#) parameter if you use per-partition scheduling. Doing so causes Moab to schedule jobs to the first partition listed, even if nodes from another partition will be available sooner.

## Grid Credential Management

- [Peer User Credential Management Overview](#)
- [Credential Mapping Files](#)

### Peer Credential Management Overview

Moab provides a number of credential management features that allow sites to control which local users can utilize remote resources and which remote users can utilize local resources and under what conditions this access is granted.

### Peer Credential Mapping

If two peers share a common user space (a given user has the same login on both clusters), then there is often no need to enable credential mapping. When users, groups, classes, QoS's, and accounts are not the same from one peer to another, Moab allows a site to specify an Object Map URL. This URL contains simple one to one or expression based mapping for credentials and other objects. Using the [RMCFG](#) parameter's **OMAP** attribute, a site can tell Moab where to find these mappings. The object map uses the following format:

```
<OBJECTTYPE>:<SOURCE_OBJECTID>,<DESTINATION_OBJECTID>
```

where *<SOURCE\_OBJECT>* can be a particular username or an asterisk (\*) which is a wildcard matching all credentials of the specified type which have not already been matched.

The object map file can be used to translate the following:

Keyword	Objects
<b>account</b>	accounts/projects
<b>class</b>	classes/queues
<b>file</b>	files/directories
<b>group</b>	groups

Keyword	Objects
<b>node</b>	nodes
<b>qos</b>	QoS
<b>user</b>	users

The following `moab.cfg` and `omap.dat` files demonstrate a sample credential mapping.

```
SCHEDCFG[master1] MODE=normal
RMCFG[slave1] OMAP=file:///opt/moab/omap.dat
...
```

```
user:joe,jsmith
user:steve,sjohnson
group:test,staff
class:batch,serial
user:*,grid
```

In this example, a job that is being migrated from cluster *master1* to the peer *slave1* will have its credentials mapped according to the contents of the `omap.dat` file. In this case, a job submitted by user *joe* on *master1* will be executed under the user account *jsmith* on peer *slave1*. Any credential that is not found in the mapping file will be passed to the peer as submitted. In the case of the user credential, all users other than *joe* and *steve* will be remapped to the user *grid* due to the wildcard matching.

Because the **OMAP** attribute is specified as a URL, multiple methods can be used to obtain the mapping information. In addition to the file protocol shown in the example above, `exec` may be used.

Note that there is no need to use the credential mapping facility to map all credentials. In some cases, a common user space exists but it is used to map all classes/queues on the source side to a single queue on the destination side. Likewise, for utilization tracking purposes, it may be desirable to map all source account credentials to a single cluster-wide account.

## Source and Destination Side Credential Mapping

Credential mapping can be implemented on the source cluster, destination cluster, or both. A source cluster may want to map all user names for all outgoing jobs to the name *generaluser* for security purposes, and a destination cluster may want to remap all incoming jobs from this particular user to the username *cluster2* and the QoS *grid*.

## Preventing User Space Collisions

In some cases, a cluster may receive jobs from two independent clusters where grid wide username distinctiveness is not guaranteed. In this case, credential mapping can be used to ensure the uniqueness of each name. With credential mapping files, this can be accomplished using the `<DESTINATION_CREDENTIAL>` wildcard asterisk (\*) character. If specified, this character will be replaced with the exact `<SOURCE_CREDENTIAL>` when generating the destination credential string. For example, consider the following configuration:

```
SCHEDCFG[master1] MODE=normal
RMCFG[slave1] OMAP=file:///opt/moab/omap.dat  FLAGS=client
...
```

```
user:*,cl_*
group:*,*_grid
account:*,temp_*
```

This configuration will remap the usernames of all jobs coming in from the peer *slave1*. The username *john* will be remapped to *c1\_john*, the group *staff* will be remapped to *staff\_grid* and the account *demo* will be remapped to *temp\_demo*.

## Grid Data Management



This method of data staging has been deprecated in Moab Workload Manager 8.0.0 and will be removed from the product in a future release. See [About data staging on page 901](#) for information about the new method of staging data.

- [Grid Data Management Overview](#)
- [Configuring Peer Data Staging](#)
- [Peer-to-Peer SCP Key Authentication](#)
- [Diagnostics](#)

### \_Grid Data Management Overview

Moab provides a highly generalized data manager interface that can allow both simple and advanced data management services to be used to migrate data amongst peer clusters. Using a flexible script interface, services such as *scp*, *NFS*, and *gridftp* can be used to address data staging needs. This feature enables a Moab peer to push job data to a destination Moab peer.

### \_Configuring Peer Data Staging

Moab offers a simple, automatic configuration, as well as advanced configuration options. At a high level, configuring data staging across a peer-to-peer relationship consists of configuring one or more storage managers, associating them with the appropriate peer resource managers, and then specifying data requirements at the local level—when the job is submitted.

To use the data staging features, you must specify the `--with-grid` option at `./configure` time. After properly configuring data staging, you can submit a job to the peer with any user who has SSH keys set up and Moab will automatically or implicitly stage back the standard out and standard error files created by the job. Files can be implicitly staged in or out before a job runs by using the [mstagein](#) or [mstageout](#) options of [msub](#).

## Simple Configuration

Moab automatically does most of the data staging configuration based on a simplified set of parameters (most common defaults) in the configuration file (`moab.cfg`).

Do the following to configure peer data staging:

1. Configure at least two Moab clusters to work in a grid. Please refer to information throughout [Moab Workload Manager for Grids](#) for help on configuring Moab clusters to work together as peers in a grid.
2. [Set up SSH keys](#) so that users on the source grid peer can SSH to destination peers without the need for a password.
3. Make necessary changes to the `moab.cfg` file of the source grid peer to activate data staging, which involves creating a new data resource manager definition within Moab. The resource manager provides data staging services to existing peers in the grid. By defining the data resource manager within the `moab.cfg`, Moab automatically sets up all of the necessary data staging auxiliary scripts.

Use the following syntax for defining a data resource manager:

```
RMCFG[<RMName>] TYPE=NATIVE RESOURCETYPE=STORAGE
VARIABLES=DATASPACEUSER=<DataSpaceUser>,DATASPACE=DIR=<DataSpaceDir>
SERVER=<DataServer>
```

- `<RMName>`: Name of the RM (defined as a storage RM type by `RESOURCETYPE=STORAGE`).
- `<DataSpaceUser>`: User used to SSH into `<DataServer>` to determine available space in `<DataSpaceDir>`. Moab runs a command similar to the following:  

```
ssh <DataServer> -l <DataSpaceUser> df <DataSpaceDir>
```
- `<DataSpaceDir>`: Directory where staged data is stored.
- `<DataServer>`: Name of the server where `<DataSpaceDir>` is located.

Define the following URLs:

```
RMCFG[data] CLUSTERQUERYURL=exec://$TOOLSDIR/cluster.query.dstage.pl
RMCFG[data] SYSTEMMODIFYURL=exec://$TOOLSDIR/system.modify.dstage.pl
RMCFG[data] SYSTEMQUERYURL=exec://$TOOLSDIR/system.query.dstage.pl
RMCFG[data] RMINITIALIZEURL=exec://$TOOLSDIR/setup.config.pl
```

4. Associate the data resource manager with a peer resource manager.

```
RMCFG[remote_data] TYPE=NATIVE RESOURCETYPE=STORAGE
VARIABLES=DATASPACEUSER=datauser,DATASPACE=DIR=/tmp SERVER=clusterhead
RMCFG[remote_cluster] TYPE=MOAB SERVER=clusterhead:42559 DATARM=remote_data
```

5. Restart Moab to finalize changes. You can use the [mschedctl -R](#) command to cause Moab to automatically restart and load the changes.

When restarting, Moab recognizes the added configuration and runs a Perl script in the Moab tool directory that configures the external scripts (also found in the tools directory) that Moab uses to perform data staging. You can view the data staging configuration by looking at the `config.dstage.pl` file in `$MOABHOMEDIR/etc`.

## Advanced Configuration

If you need a more customized data staging setup, contact your account representative.

### Peer-to-Peer SCP Key Authentication

In order to use scp as the data staging protocol, we will need to create SSH keys which allow users to copy files between the two peers, without the need for passwords. For example, if *UserA* is present on the source peer, and his counterpart is *UserB* on the destination peer, then *UserA* will need to create an SSH key and configure *UserB* to allow password-less copying. This will enable *UserA* to copy files to and from the destination peer using Moab's data staging capabilities.

Another common scenario is that several users present on the source peer are mapped to a single user on the destination peer. In this case, each user on the source peer will need to create keys and set them up with the user at the destination peer. Below are steps that can be used to setup SSH keys among two (or more) peers:

**i** These instructions were written for [OpenSSH version 3.6](#) and might not work correctly for older versions.

#### Generate SSH Key on Source Peer

As the user who will be submitting jobs on the source peer, run the following command:

```
ssh-keygen -t rsa
```

You will be prompted to give an optional key. Just hit return and ignore this or other settings. When finished, this command will create two files `id_rsa` and `id_rsa.pub` located inside the user's `~/.ssh/` directory.

#### Copy the Public SSH Key to the Destination Peer

Transfer the newly created public key (`id_rsa.pub`) to the destination peer:

```
scp ~/.ssh/id_rsa.pub ${DESTPEERHOST}:~
```

#### Disable Strict SSH Checking on Source Peer (Optional)

By appending the following to your `~/.ssh/config` file you can disable SSH prompts which ask to add new hosts to the "known hosts file." (These prompts can often cause problems with data staging functionality.) Note that the `${DESTPEERHOST}` should be the name of the host machine running the destination peer:

```
Host ${DESTPEERHOST}
  CheckHostIP no
  StrictHostKeyChecking no
  BatchMode yes
```

#### Configure Destination Peer User

Now, log in to the destination peer as the destination user and set up the newly created public key to be trusted:



**i** The number of bytes transferred for each file is currently not used.

```
> checknode -v scp://keche//tmp/
node scp://keche//tmp/
State:      Idle (in current state for 00:00:13)
Configured Resources: DISK: 578G
Utilized Resources: DISK: 316G
Dedicated Resources: ---
  MTBF(longterm):  INFINITY  MTBF(24h):  INFINITY
Active Data Staging Operations:
  job      native.2 complete (1 bytes transferred)
(/home/brian/stage.txt)
  job      native.3 pending (1 bytes) (/home/brian/stage.txt)
Dedicated Storage Manager Disk Usage:  0 of 592235 MB
Cluster Query URL:  exec://$TOOLS DIR/grid/cluster.query.dstage.pl
Partition:  SHARED Rack/Slot:  ---
Flags:      rmdetected
RM[data]:   TYPE=NATIVE
EffNodeAccessPolicy: SHARED
Total Time: 00:12:15  Up: 00:12:15 (100.00%)  Active: 00:00:00 (0.00%)
Reservations:  ---
```

- **mdiag -n**: Displays the state of the storage node.

```
> mdiag -n
compute node summary
Name                State  Procs   Memory      Opsys
compute1            Idle   4:4     3006:3006   linux
compute2            Down   0:4     3006:3006   linux
scp://keche//tmp/   Idle   0:0     0:0         -
-----            ---    4:8     6012:6012   -----
Total Nodes: 3 (Active: 0 Idle: 2 Down: 1)
```

- **checkjob -v**: Displays the status of the staging request.

**i** The remaining time and size of the file information is currently not used. The information should only be used to see file locations and whether the file has been staged or not.

```
> checkjob -v jobid
...
Stage-In Requirements:
  localhost:/home/brian/stage.txt => keche://tmp/staged.txt  size:0B
status:[NONE]  remaining:00:00:01
  Transfer URL: file:///home/brian/stage.txt,ssh://keche/tmp/staged.txt
...
```

To ensure that SCP key authentication is properly configured, the following conditions must be met:

- Moab is running as root.
- You are able to issue the following command as the root user without being prompted for a password:

```
su - <DATASPACEUSER> -c "/usr/bin/ssh <destination host> -l <DATASPACEUSER> 'df
-k //tmp/ 2>&1 || echo FAILED'"
```

- You can SSH *<destination host>* without a password.

- The `dataSpaceLocalUser` and `dataSpaceMappedUser` variables in your `/opt/moab/etc/config.dstage.pl` script are set to the same username you assigned through `<DATASPACEUSER>`.

## Grid Security

### Secret Key Based Server Authentication

Secret key based security is required in order for the grid to work. It is enabled in the `moab-private.cfg` file. Configuration of `moab-private.cfg` is covered throughout the grid configuration documentation, as well as in [Appendix E: Security](#).

## Grid Diagnostics and Validation

- [Peer Management Overview](#)
- [Peer Diagnostic Overview](#)

### Peer Management Overview

- Use `mdiag -R` to view interface health and performance/usage statistics.
- Use `mrmctl` to enable/disable peer interfaces.
- Use `mrmctl -m` to dynamically modify/configure peer interfaces.

### Peer Diagnostic Overview

- Use `mdiag -R` to diagnose general RM interfaces.
- Use `mdiag -S` to diagnose general scheduler health.
- Use `mdiag -R -V job <RMID>` to diagnose peer-to-peer job migration.

```
> mdiag -R -V job peer1
```

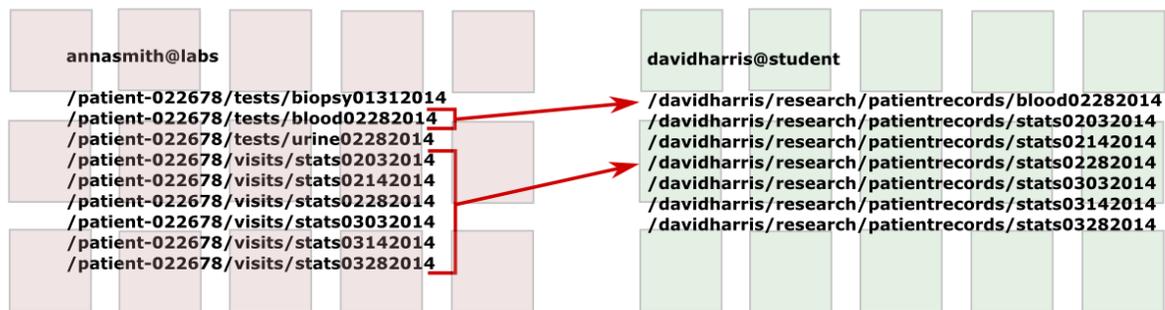
- Use `mdiag -R -V data <RMID>` to diagnose peer-to-peer data staging.
- Use `mdiag -R -V cred <RMID>` to diagnose peer-to-peer credential mapping.

# Data staging

## About data staging

Sometimes you might need a job to process data that resides at another site. With the proper configuration, you can submit your job with the requirement that it copies data from the external site to yours and, if needed, copy the job's resulting data out to the external site for its owner to use. Data staging is an out-of-band method of moving data without reserving compute nodes or other resources for it.

In the example below, which will appear throughout the chapter, a university researcher needs the results of tests done at a hospital. User  `davidharris`  on the  `student`  server of the university submits a job called  `Moab . 1`  that requires several files stored by user  `annasmith`  on the research server of the hospital.  `davidharris`  submitted  `Moab . 1`  with certain options in place that instruct Moab to copy the files to the  `/student/davidharris/research/patientrecords/`  directory on the  `student`  server prior to starting the job.



Moab currently supports the following data staging use cases: 1) Staging data to or from a shared file system, 2) Staging data to or from local node storage on a single compute node, and 3) Staging data to or from a shared file system on an unspecified cluster – resolved at job migration – in a grid configuration.

Before you can submit data staging jobs, you must configure certain generic metrics in your partitions, job templates, and the data staging submit filter for data staging scheduling, throttling, and policies.

Moab uses Linux file transfer utilities to stage the data and includes data staging reference scripts that support the  `scp`  and  `rsync`  Linux file transfer utilities. The scripts will work for standard installations, but you can customize the script to support data staging to and from an external staging server, the Moab server itself, or a local compute node, depending on your implementation. You can also customize your own script for other file transfer utilities, such as  `Aspera` .

Once you configure your system to support data staging, you can begin creating data staging jobs by attaching the  `--stagein` ,  `--stageinfile` ,  `--stageinsize` ,  `--stageout` ,  `--stageoutfile` , and  `--stageoutsize`  options to your  `msub`  commands. See [Staging data on page 299](#) for more information.

## How-to's

The following topics describe how to stage data in different Moab environments.

- [Configuring the SSH keys for the data staging transfer script on page 903](#)
- [Configuring data staging on page 905](#)
- [Configuring the \\$CLUSTERHOST variable on page 915](#)
- [Staging data to or from a shared file system on page 907](#)
- [Staging data to or from a shared file system in a grid on page 911](#)
- [Staging data to or from a compute node on page 916](#)
- [Configuring data staging with advanced options on page 920](#)

## References

The following topics contain detailed information that you can use as reference material for data staging

- [Sample user job script on page 922](#)
- [Applying the msub submit filter on page 303](#)

## Related topics

- [msub on page 286](#)

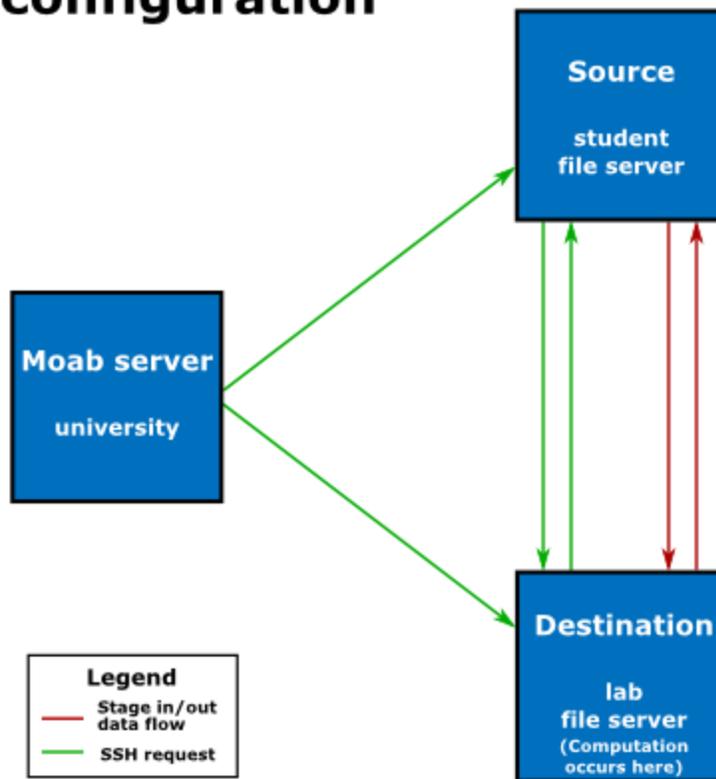
## How-to's

### Configuring the SSH keys for the data staging transfer script

#### Context

For data staging to work correctly, you must configure SSH keys to allow the data staging scripts to run without passphrases. In the sample data staging server configuration shown in the image below, davidharris on the student server stages data from the source server student to the destination server labs. The computation occurs on the labs server before Moab stages the output data from labs back to student. The image below demonstrates the SSH connections necessary and how you should configure your SSH keys.

### Data staging server configuration



For more details on generating keys, see the [ssh-keygen man page](#) and "[SSH login without password](#)".

## To configure the SSH keys for the data staging transfer script

1. Generate a new SSH key on the Moab server (`university`) if one does not already exist. To do so, run each of the following steps.

- a. Run `ssh-keygen` to generate a public and private rsa key pair.

```
[davidharris@university]$ ssh-keygen
```

- b. Enter the name of the file where you want to store the key, or you can accept the default location.

```
/home/davidharris/.ssh/id_rsa
```

- c. When prompted for a passphrase, leave it blank and press **Enter**. Repeat when prompted to retype passphrase.

2. Install the public key on the source and destination hosts. Note that in this example the source host is `student` and the destination host is `labs`.

- a. Copy the `university` public key to `student`. Answer **yes** to continue connecting.

```
[davidharris@university]$ ssh-copy-id -i ~/.ssh/id_rsa.pub student
```

- b. Copy the `university` public key to `labs`. Answer **yes** to continue connecting.

```
[davidharris@university]$ ssh-copy-id -i ~/.ssh/id_rsa.pub labs
```



The next two steps generate a key-pair for each node. It is acceptable to generate a single key-pair and install it on each node. It does not matter where the key-pair is generated, so long as it is compatible with the SSH client/server.

3. Generate a key pair on the source host (`student`) and install the public key generated to the destination host (`labs`). When prompted for a passphrase, leave it blank and press **Enter**. Repeat when prompted to retype passphrase.

```
[davidharris@student]$ ssh-keygen
[davidharris@student]$ ssh-copy-id -i ~/.ssh/id_rsa.pub labs
```

4. Generate a key pair on the destination host (`labs`) and install the public key generated to the source host (`student`). When prompted for a passphrase, leave it blank and press **Enter**. Repeat when prompted to retype passphrase.

```
[davidharris@labs]$ ssh-keygen
[davidharris@labs]$ ssh-copy-id -i ~/.ssh/id_rsa.pub student
```

5. Ensure that each user who will run data staging jobs has read and write permissions on each source and destination server.

6. Test the configuration. To do so:

- a. Install the modules required to run the data staging scripts. `python-paramiko` is required for data staging, but `python-mock` is only required if you intend to run the unit test.

```
> yum install python-paramiko python-mock
```

- b. Transfer a file from the source host to the destination host to verify that the keys work for the users configured. To do so, run `/opt/moab/tools/data-staging/ds_move_scp --test=<source>%<destination>` if you use `scp` or `/opt/moab/tools/data-staging/ds_move_rsync --test=<source>%<destination>` script if you use `rsync`. `<source>%<destination>` is configured the same way as the `--stagein` and `--stageout` options for `msub`; for help configuring your source and destination, see [Staging a file or directory on page 299](#).

```
[davidharris@university]$ /opt/moab/tools/data-staging/ds_move_rsync --test=davidharris@student:/tmp/test%davidharris@labs:/tmp
```

- c. In the same way, transfer a file from the destination host to the source host to verify that the keys work for the users configured.

```
[davidharris@university]$ /opt/moab/tools/data-staging/ds_move_rsync --test=davidharris@labs:/tmp/test%davidharris@student:/test_processed
```

## Related topics

- [About data staging on page 901](#)
- [Configuring data staging on page 905](#)

## Configuring data staging

### Context

You must modify your Moab configuration to enable data staging. In addition to the configuration steps described below, you might also consider customizing the configuration (including the associated scripts) to meet your site's specific needs.

For advanced configuration steps and options, see [Configuring data staging with advanced options on page 920](#).

### To configure data staging

1. Verify that your firewall and network are correctly configured to allow the scripts to operate as designed.
2. If you have not already done so, install the modules required to run the data staging scripts. `python-paramiko` is required for data staging, but `python-mock` is only required if you intend to run the unit test.

```
> yum install python-paramiko python-mock
```

3. If you have not already, follow the instructions found in [Configuring the SSH keys for the data staging transfer script on page 903](#).
4. Ensure that the data staging scripts are installed on your system. To do so, list the contents of the `/opt/moab/tools/data-staging` directory. You should see the data staging README file, reference scripts, and other related files.

```
> ls -l /opt/moab/tools/data-staging
```

You can copy and modify the reference scripts and configuration files to meet your specific needs. See the README file packaged in the `data-staging` directory for information about modifying these files.

5. Open your `moab.cfg` file for editing and do each of the following tasks:
  - a. Configure the data staging `msub` filter, located in `/opt/moab/tools/data-staging` by default, as a client-side filter. See [Applying the msub submit filter on page 303](#) for more information.

```
SUBMITFILTER /opt/moab/tools/data-staging/ds_filter
```

The data staging filter checks the `msub` argument syntax to verify that the arguments make sense and are consistent; attempts a dry run connection via SSH and the file transfer utility to ensure that keys exist for the user on the necessary systems; and attempts to determine the size of the data that will be transferred.

You can customize the script to meet your specific needs; the file contains detailed comments illustrating its default behavior to facilitate its modification. If you replace or modify the submit filter, it is your responsibility to ensure that the same functionality described in the paragraph above is present in your filter.

Note that this filter has the `DEFAULT_TEMPLATE` name which should match the name of the master data staging template in `moab.cfg`. For more information, see [Configuring data staging with advanced options on page 920](#).

- b. Set the data staging bandwidth gmetric (`DATASTAGINGBANDWIDTH_MBITS_PER_SEC`) on each partition associated with an RM to the rate at which its network to be used for data staging transfers data in megabits per second (see [Per-Partition Settings on page 497](#) for more information). Moab will use the specified rate and the data staging size specified at job submission (see [Stage in or out file size on page 301](#) for more information) to determine how long staging the data will take and to schedule the job as soon after data staging completes as possible.

#### Example 3-175: Non-grid

```
RMCFG[torque] Type=pbs
PARCFG[torque] GMETRIC[DATASTAGINGBANDWIDTH_MBITS_PER_SEC]=58
```

*Partition `torque` has a transfer rate of 58 megabits per second. Moab uses the rate when it estimates the time it will take to stage data in and determine when to schedule the job that will use the data.*

#### Example 3-176: Grid

```
RMCFG[m1] type=Moab
PARCFG[m1] GMETRIC[DATASTAGINGBANDWIDTH_MBITS_PER_SEC]=100
```

*Partition `m1` has a transfer rate of 100 megabits per second. Moab uses the rate when it estimates the time it will take to stage data in and determine when to schedule the job that will use the data.*

- c. Set the bandwidth generic resource on all nodes to limit the total number of concurrent data staging jobs in your system.

```
NODECFG[GLOBAL] GRES=bandwidth:10
```

*Data staging jobs can use up to 10 units of bandwidth on the system. You can specify the number of units consumed by each data staging job when you configure the data staging job templates.*

- 6. Install the msub client filter on all client submission hosts.

Related topics

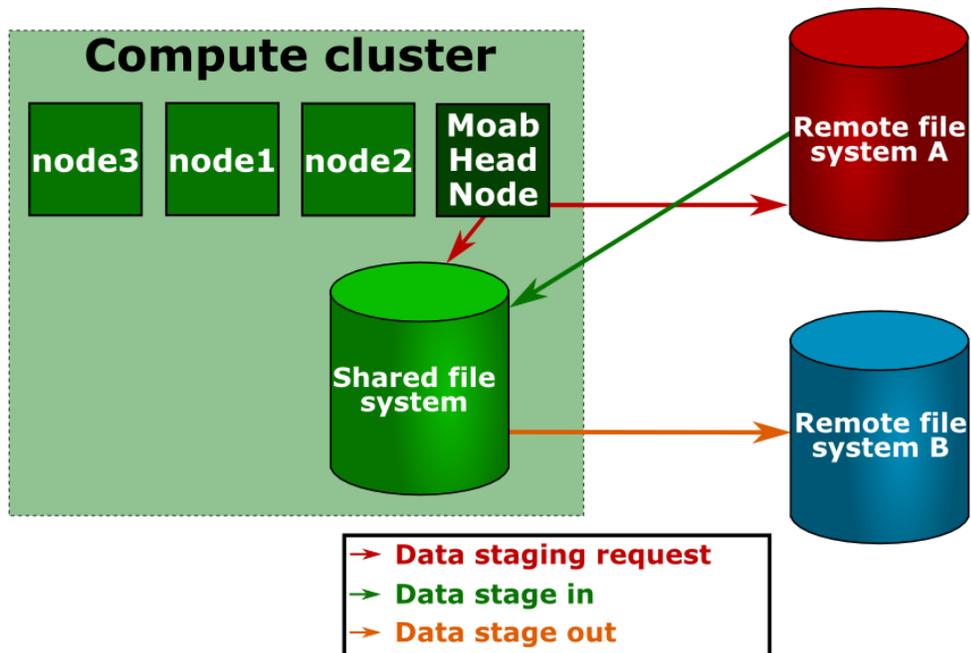
- [About data staging on page 901](#)

## Staging data to or from a shared file system

Context

In the most common data staging use case, the cluster utilizes a shared file system between all compute nodes. This type of data staging makes data stored outside of the cluster available to a job that will run on any set of nodes in the cluster. At the time of submission, you must specify where Moab will obtain the data with a username, host name, and path to a file or directory and where on the shared file system Moab will store the data. After the job runs, you can also copy data from the shared file system back to a remote file system.

Image 3-13: Data staging to or from a shared file system



## To stage data to or from a shared file system

1. If you have not already done so, configure your SSH keys and `moab.cfg` to support data staging. See [Configuring the SSH keys for the data staging transfer script on page 903](#) and [Configuring data staging on page 905](#) for more information.
2. Create your job templates for data staging jobs in `moab.cfg`. The templates in the example below create a compute job that stages data in before it starts and stages data out when it completes. For more information about creating job templates, see [About job templates on page 853](#).
  - a. Create a selectable master template, called `ds` in the example below, that creates a stage in and stage out system job. This name should match the `DEFAULT_TEMPLATE` value in `ds_config.py`. See [Configuring data staging with advanced options on page 920](#) for more information.
  - b. For the data staging in job template, called `dsin` in the example below, specify that it will create a data staging job by setting `DATASTAGINGJOB` to `TRUE`. Note that the name of this job template must match the name of the data stage in job template referenced in the master template.
  - c. Set the bandwidth `GRES` to the amount of bandwidth a single stage in job should use. This indicates how many of the bandwidth units specified with `NODECFG[GLOBAL]` in [Configuring data staging on page 905](#) a data staging job with this template should consume.
  - d. Add `FLAGS=GRESONLY` to indicate that this data staging job does not require any compute resources.
  - e. Create a trigger that executes the `ds_move_scp`, `ds_move_rsync`, or [ds move multiplex](#) script, depending on which file transfer utility you use. Set the `attacherror`, `objectxmlstdin`, and `user` `FLAGS` to attach any trigger stderr as a message to the job, pass the job XML to the script, and indicate that the script should run as the job's user, respectively.

**i** If you use the `rsync` protocol, you can configure your data staging jobs to report the actual number of bytes transferred and the total data size to be transferred. To do so, use the `Sets` attribute to `^BYTES_IN.^DATA_SIZE_IN` for stage in jobs and `^BYTES_OUT.^DATA_SIZE_OUT` for stage out jobs. For example, a stage in trigger would look like the following:

```
JOBCFG[dsin]
TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_move_rsync --stagein",Flags=objectxmlstdin:user:attacherror,Sets=^BYTES_IN.^DATA_SIZE_IN
```

A stage out trigger would look like the following:

```
JOBCFG[dsout]
TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_move_rsync --stageout",Flags=objectxmlstdin:user:attacherror,Sets=^BYTES_OUT.^DATA_SIZE_OUT
```

These variables show up as events if you set your `WIKIEVENTS` parameter to `TRUE`.

- f. Create the stage out job, called `dsout` in the example below, by repeating steps 2b - 2e in a new template. In the example below, this template is called `dsout`. Note that the name of this job

template must match the name of the data stage out job template referenced in the data staging master template.

```

JOBCFG[ds]      TEMPLATEDEPEND=AFTEROK:dsin  TEMPLATEDEPEND=BEFORE:dsout
SELECT=TRUE

JOBCFG[dsin]   DATASTAGINGSYSJOB=TRUE
JOBCFG[dsin]   GRES=bandwidth:2
JOBCFG[dsin]   FLAGS=GRESONLY
JOBCFG[dsin]   TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_rsync --stagein",Flags=attacherror:objectxmlstdin:user

JOBCFG[dsout]  DATASTAGINGSYSJOB=TRUE
JOBCFG[dsout]  GRES=bandwidth:1
JOBCFG[dsout]  FLAGS=GRESONLY
JOBCFG[dsout]  TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_rsync --stageout",Flags=attacherror:objectxmlstdin:user

```

3. Create the job using `msub`, adding resources and specifying a script as you normally would. Then configure Moab to stage the data for it. To do so:

- a. At the end of the command, use the `--stagein/--stageout` option and/or `--stageinfile/--stageoutfile` option.
  - The `--stagein/--stageout` option lets you specify a single file or directory to stage in or out. You must set the option equal to `<source>%<destination>`, where `<source>` and `<destination>` are both `[<user>@]<host>:/<path>/[<fileName>]`. See [Staging a file or directory on page 299](#) for format and details.

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

```
> msub --stagein=annasmith@labs:/patient-
022678/%davidharris@university:/davidharris/research/patientrecords
<jobScript>
```

*Moab copies the /patient-022678 directory from the hospital's labs server to the university cluster where the job will run prior to job start.*

- The `--stageinfile/--stageoutfile` option lets you specify a file that contains the file and/or directory name(s) to stage in or out. You must set the option equal to `<path>/<fileName>` of the file. The file must contain at least one line with this format: `<source>%<destination>`, where both `<source>` and `<destination>` are `[<user>@]<host>:/<path>/[<fileName>]`. See [Staging multiple files or directories on page 300](#) for more information.

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

```
> msub --stageinfile=/davidharris/research/recordlist <jobScript>
```

*Moab copies all files specified in the /davidharris/research/recordlist file to the cluster where the job will run prior to job start.*

```
/davidharris/research/recordlist:
```

```
annasmith@labs:/patient-
022678/tests/blood02282014%davidharris@university:/davidharris/research/patie
ntrecords/blood02282014
annasmith@labs:/patient-
022678/visits/stats02032014%davidharris@university:/davidharris/research/pati
entrecords/stats02032014
annasmith@labs:/patient-
022678/visits/stats02142014%davidharris@university:/davidharris/research/pati
entrecords/stats02142014
annasmith@labs:/patient-
022678/visits/stats02282014%davidharris@university:/davidharris/research/pati
entrecords/stats02282014
annasmith@labs:/patient-
022678/visits/stats03032014%davidharris@university:/davidharris/research/pati
entrecords/stats03032014
annasmith@labs:/patient-
022678/visits/stats03142014%davidharris@university:/davidharris/research/pati
entrecords/stats03142014
annasmith@labs:/patient-
022678/visits/stats03282014%davidharris@university:/davidharris/research/pati
entrecords/stats03282014
```

*Moab copies the seven patient record files from the hospital's labs server to the university cluster where the job will run prior to job start.*

- b. The `--stageinsize/--stageoutsize` option lets you specify the estimated size of the files and/or directories to help Moab more quickly and accurately calculate the amount of time it will take to stage the data and therefore schedule your job correctly. If you are staging data out, then setting `--stageoutsize` is required. If you provide an integer, Moab will assume the number is in megabytes. To change the unit, add another suffix. See [Stage in or out file size on page 301](#) for more information.

```
> msub --stageinfile=/davidharris/research/recordlist --stageinsize=100
<jobScript>
```

*Moab copies the /davidharris/research/recordlist file, which is approximately 100 megabytes, from the biology node to the host where the job will run prior to job start.*

4. To see the status, errors, and other details associated with your data staging job, run `checkjob -v`. See "[checkjob](#)" for details.

#### Related topics

- [About data staging on page 901](#)
- [Configuring data staging on page 905](#)
- [Configuring data staging with advanced options on page 920](#)
- [Staging data to or from a shared file system in a grid on page 911](#)
- [Staging data to or from a compute node on page 916](#)
- [Sample user job script on page 922](#)

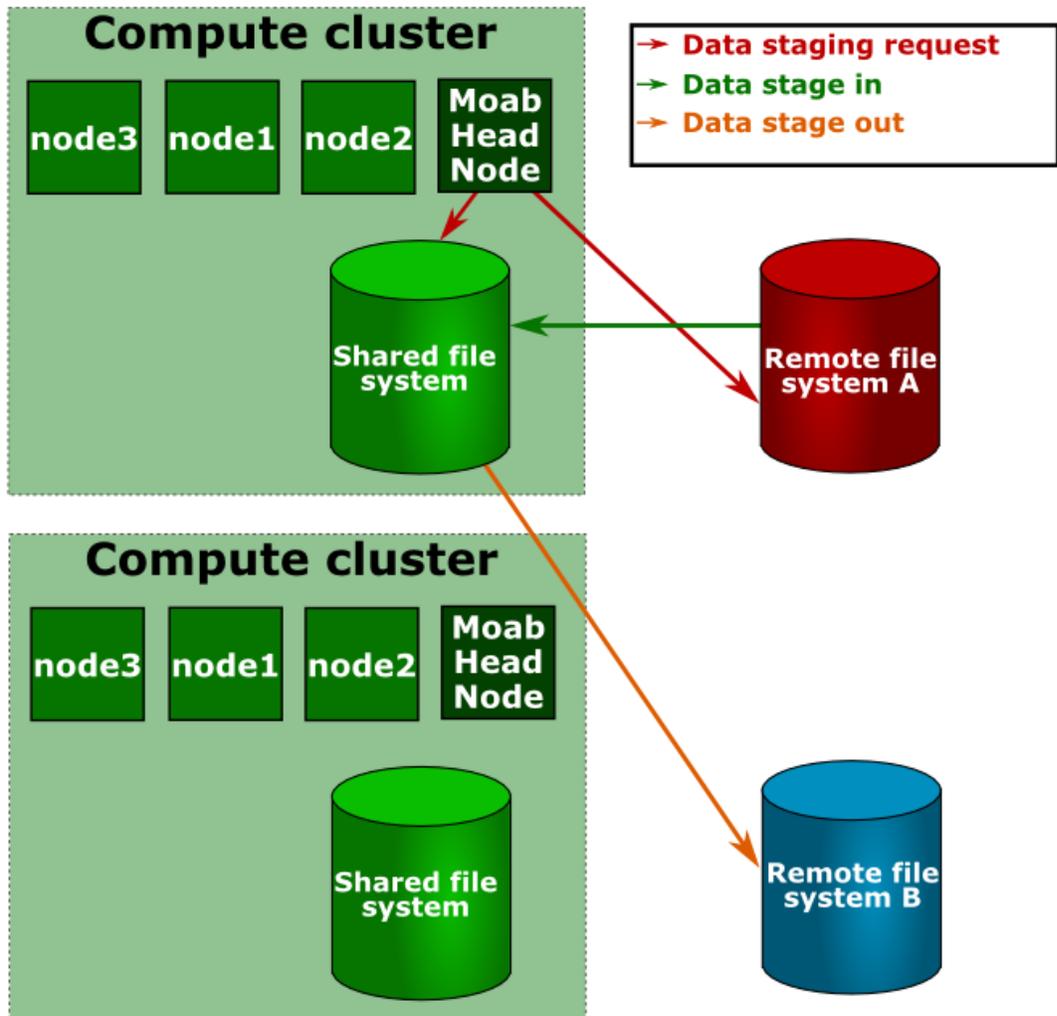
## Staging data to or from a shared file system in a grid

### Context

You can stage data in an environment where multiple instances of Moab run in a grid configuration. For this type of data staging, each cluster utilizes a shared file system with all compute nodes. This type of data staging will make data available to a job that will run on a set of nodes in one of the clusters in the grid. You must specify where the remote data can be obtained with a username, host name, and path to a file or directory and where on the shared storage Moab will store the data. The remote data source location is known at job submission time, but you must use the `$CLUSTERHOST` placeholder for the host name of the data transfer server on which the job will be scheduled. After the job runs, you can also copy data from the cluster shared file system to a remote file system.

Note that you cannot stage data to or from a local compute node with its own local storage in a grid environment.

Image 3-14: Data staging in a grid



## To stage data to or from a shared file system in a grid

1. If you have not already done so, configure your SSH keys and `moab.cfg` to support data staging. See [Configuring the SSH keys for the data staging transfer script on page 903](#) and [Configuring data staging on page 905](#) for more information.
2. Create your job templates for data staging jobs in `moab.cfg`. The templates in the example below create a compute job that stages data in before it starts and stages data out when it completes. For more information about creating job templates, see [About job templates on page 853](#).
  - a. Create a selectable master template, called `ds` in the example below, that creates a stage in and stage out system job. This name should match the `DEFAULT_TEMPLATE` value in `ds_config.py`. For more information, see [Configuring data staging with advanced options on page 920](#).
  - b. For the data staging in job template, called `dsin` in the example below, specify that it will create a data staging job by setting `DATASTAGINGJOB` to `TRUE`. Note that the name of this job template must match the name of the data stage in job template referenced in the master template.
  - c. Set the staging job template bandwidth `GRES` to the amount of bandwidth a single stage in job should use. This indicates how many of the bandwidth units specified with `NODECFG[GLOBAL]` in [Configuring data staging on page 905](#) a data staging job with this template should consume.
  - d. Set `JOBMIGRATEPOLICY` to `JUSTINTIME`.
  - e. Add `FLAGS=GRESONLY` to indicate that this data staging job does not require any compute resources.
  - f. Create a trigger that executes the `ds_move_scp`, `ds_move_rsync`, or `ds_move_multiplex` script, depending on which file transfer utility you use. Set the `attacherror`, `objectxmlstdin`, and `user` `FLAGS` to attach any trigger stderr as a message to the job, pass the job XML to the script, and indicate that the script should run as the job's user, respectively.

**i** If you use the `rsync` protocol, you can configure your data staging jobs to report the actual number of bytes transferred and the total data size to be transferred. To do so, use the `Sets` attribute to `^BYTES_IN.^DATA_SIZE_IN` for stage in jobs and `^BYTES_OUT.^DATA_SIZE_OUT` for stage out jobs. For example, a stage in trigger would look like the following:

```
JOBCFG[dsin]
TRIGGER=EType=start, AType=exec, Action="/opt/moab/tools/data-staging/ds_move_rsync --stagein", Flags=objectxmlstdin:user:attacherror, Sets=^BYTES_IN.^DATA_SIZE_IN
```

A stage out trigger would look like the following:

```
JOBCFG[dsout]
TRIGGER=EType=start, AType=exec, Action="/opt/moab/tools/data-staging/ds_move_rsync --stageout", Flags=objectxmlstdin:user:attacherror, Sets=^BYTES_OUT.^DATA_SIZE_OUT
```

These variables show up as events if you set your `WIKIEVENTS` parameter to `TRUE`.

- g. Create the stage out job, called `dsout` in the example below, by repeating steps 2b - 2f in a new template. In the example below, this template is called `dsout`. Note that the name of this job template must match the name of the data stage out job template referenced in the master template.

```

JOBCFG[ds]      TEMPLATEDEPEND=AFTEROK:dsin  TEMPLATEDEPEND=BEFORE:dsout
SELECT=TRUE

JOBCFG[dsin]   DATASTAGINGSYSJOB=TRUE
JOBCFG[dsin]   GRES=bandwidth:2
JOBCFG[dsin]   FLAGS=GRESONLY
JOBCFG[dsin]   TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_rsync --stagein",Flags=attacherror:objectxmlstdin:user

JOBCFG[dsout]  DATASTAGINGSYSJOB=TRUE
JOBCFG[dsout]  FLAGS=GRESONLY
JOBCFG[dsout]  INHERITRES=TRUE
JOBCFG[dsout]  TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_rsync --stageout",Flags=attacherror:objectxmlstdin:user

```

3. Create the job using `msub`, adding resources and specifying a script as you normally would. Then configure Moab to stage the data for it. To do so:

- a. At the end of the command, use the `--stagein/--stageout` option and/or `--stageinfile/--stageoutfile` option.

- The `--stagein/--stageout` option lets you specify a single file or directory to stage in or out. You must set the option equal to `<source>%<destination>`, where `<source>` and `<destination>` are both `[<user>@] <host>:/<path>/[<fileName>]`. See [Staging a file or directory on page 299](#) for format and details.

Note that if you do not know the cluster where the job will run but want the data staged to the same location, you can use the `$CLUSTERHOST` variable in place of a host. If you choose to use the `$CLUSTERHOST` variable, you must first customize the `ds_config.py` file. For more information, see [Configuring the \\$CLUSTERHOST variable on page 915](#).

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

```
> msub ... --stagein=annasmith@labs:/patient-022678/%\${CLUSTERHOST}/davidharris/research/patientrecords <jobScript>
```

*Moab copies the /patient-022678 directory from the hospital's labs server to the cluster where the job will run prior to job start.*

- The `--stageinfile/--stageoutfile` option lets you specify a file that contains the file(s) and directory(-ies) to stage in or out. You must set the option equal to `<path>/<fileName>` of the file. The file must contain at least one line with this format: `[<user>@] <host>:/<path>[<fileName>]`. See [Staging multiple files or directories on page 300](#) for more information.

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

```
> msub ... --stageinfile=/davidharris/research/recordlist <jobScript>
```

*Moab copies all files specified in the /davidharris/research/recordlist file to the cluster where the job will run prior to job start.*

```
/davidharris/research/recordlist:
```

```
annasmith@labs:/patient-022678/tests/blood02282014%$CLUSTERHOST:/davidharris/research/patientrecords/blood02282014
annasmith@labs:/patient-022678/visits/stats02032014%$CLUSTERHOST:/davidharris/research/patientrecords/stats02032014
annasmith@labs:/patient-022678/visits/stats02142014%$CLUSTERHOST:/davidharris/research/patientrecords/stats02142014
annasmith@labs:/patient-022678/visits/stats02282014%$CLUSTERHOST:/davidharris/research/patientrecords/stats02282014
annasmith@labs:/patient-022678/visits/stats03032014%$CLUSTERHOST:/davidharris/research/patientrecords/stats03032014
annasmith@labs:/patient-022678/visits/stats03142014%$CLUSTERHOST:/davidharris/research/patientrecords/stats03142014
annasmith@labs:/patient-022678/visits/stats03282014%$CLUSTERHOST:/davidharris/research/patientrecords/stats03282014
```

*Moab copies the seven patient record files from the hospital's labs server to the cluster where the job will run prior to job start.*

- b. The `--stageinsize/--stageoutsize` option lets you specify the estimated size of the files and/or directories to help Moab more quickly and accurately calculate the amount of time it will take to stage the data and therefore schedule your job correctly. If you used the `$CLUSTERHOST` variable to stage in, then setting `--stageinsize` is required. `--stageoutsize` is always required for staging data out. If you provide an integer, Moab will assume the number is in megabytes. To change the unit, add another suffix. See [Stage in or out file size on page 301](#) for more information.

```
> msub ... --stageinfile=/davidharris/research/recordlist --stageinsize=100 <jobScript>
```

*Moab copies the /davidharris/research/recordlist file, which is approximately 100 megabytes, from the biology node to the host where the job will run prior to job start.*

4. To see the status, errors, and other details associated with your data staging job, run `checkjob -v`. See "[checkjob](#)" for details.

#### Related topics

- [About data staging on page 901](#)
- [Configuring data staging on page 905](#)

- [Configuring data staging with advanced options on page 920](#)
- [Sample user job script on page 922](#)

## Configuring the \$CLUSTERHOST variable

### Context

When you submit a data staging job in a grid environment, you can use the `$CLUSTERHOST` variable instead of specifying a destination if you do not know the cluster where the job will run but want the data staged to the same location. Before the variable will work correctly, you must first configure it by customizing the `ds_config.py` script to match your unique system.

#### Use case

In a grid with three clusters, you have a partition named `master` where you want all data staged to a host named `gridheadNAS`; a partition named `csdept` where you want all data staged to a host named `fs001.cs.example.edu`; and a partition named `lab` where you want all data staged to a host named `bigfilesystem`.

## To configure the \$CLUSTERHOST variable

1. Open the `ds_config.py` file for modification. It is located in `/opt/moab/tools/data-staging/` by default.

```
[moab]$ vi /opt/moab/tools/data-staging/ds_config.py
```

2. Locate the `PARTITION_TO_HOST` parameter.

```
...
PARTITION_TO_HOST = {"partition_1_name": "cluster_1_staging_hostname",
                    "partition_2_name": "cluster_2_staging_hostname",
                    "partition_3_name": "cluster_3_staging_hostname"}
...
```

3. Replace the partition names and associated cluster hostnames with those that match your system. For the use case provided above, you would customize it the following way:

```
...
PARTITION_TO_HOST = {"master": "gridheadNAS",
                    "csdept": "fs001.cs.example.edu",
                    "lab": "bigfilesystem"}
...
```

### Related topics

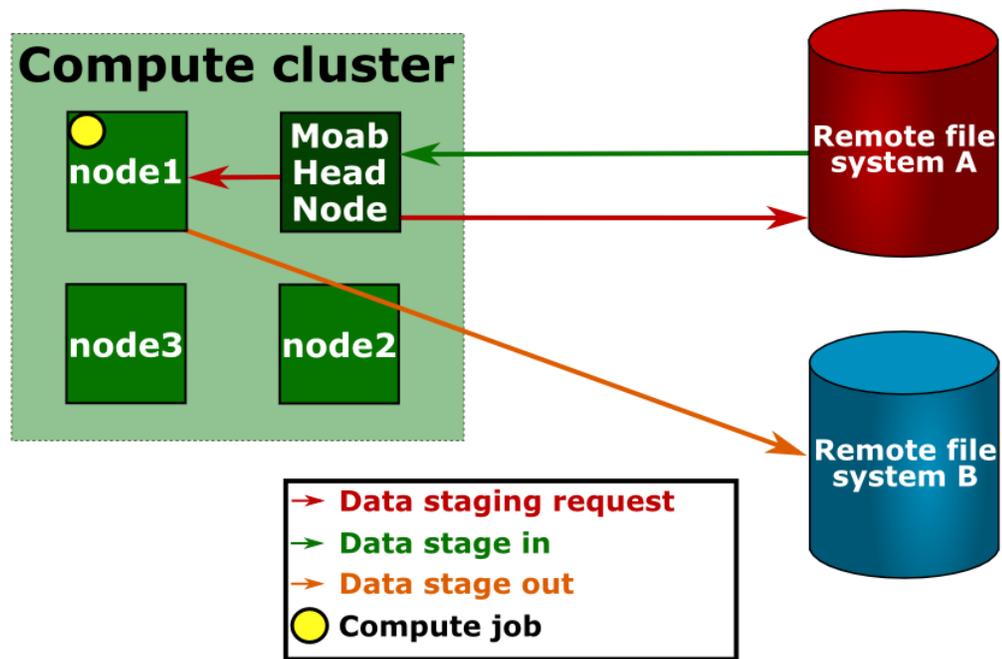
- [Staging data to or from a shared file system in a grid on page 911](#)

## Staging data to or from a compute node

### Context

You can stage data to or from a local compute node in an environment where each node on the cluster has local storage. This type of data staging will make data stored outside the cluster available to a job that will run on a single node in the cluster. You must specify the username, host name, and path to a file or directory and a location on the compute node where Moab will store the data. You will supply the remote data source location at job submission time, but you must use the `$JOBHOST` placeholder for the name of the compute node. After the job runs, you can also copy data from the local file system to a remote file system.

Image 3-15: Data staging to or from a local compute node



Before staging data to or from a local compute node, please follow the procedure in [Configuring data staging on page 905](#).

### To stage data to or from a local compute node

1. If you have not already done so, configure your SSH keys and `moab.cfg` to support data staging. See [Configuring the SSH keys for the data staging transfer script on page 903](#) and [Configuring data staging on page 905](#) for more information.
2. Create your job templates for data staging jobs in `moab.cfg`. The templates in the example below create a compute job that stages data in before it starts and stages data out when it completes. For more information about creating job templates, see [About job templates on page 853](#).
  - a. Create a selectable master template, called `ds` in the example below, that creates a stage in and stage out system job. This name should match the `DEFAULT_TEMPLATE` value in `ds_config.py`.

For more information, see [Configuring data staging with advanced options on page 920](#).

- b. For the data staging in job template, called `dsin` in the example below, specify that it will create a data staging job by setting **DATASTAGINGJOB** to *TRUE*. Note that the name of this job template must match the name of the data stage in job template referenced in the master template.
- c. Set the staging job template bandwidth **GRES** to the amount of bandwidth a single stage in job should use. This indicates how many of the bandwidth units specified with **NODECFG** [GLOBAL] in [Configuring data staging on page 905](#) a data staging job with this template should consume.
- d. For local node data staging it is important that the data staging job has the entire node to itself. To prevent Moab from scheduling another job on the node at the same time as the data staging job, set the **NODEACCESSPOLICY** to *SINGLEJOB* in the staging job template.
- e. Add **INHERITRES=TRUE** to reserve the compute node for the data staging job to prevent other compute jobs from using the node at the same time and creating input, output, and disk conflicts with the data staging job.
- f. Create a trigger that executes the `ds_move_scp`, `ds_move_rsync`, or `ds_move_multiplex` script, depending on which file transfer utility you use. Set the *attacherror*, *objectxmlstdin*, and *user* **FLAGS** to attach any trigger stderr as a message to the job, pass the job XML to the script, and indicate that the script should run as the job's user, respectively.

**i** If you use the rsync protocol, you can configure your data staging jobs to report the actual number of bytes transferred and the total data size to be transferred. To do so, use the **Sets** attribute to `^BYTES_IN.^DATA_SIZE_IN` for stage in jobs and `^BYTES_OUT.^DATA_SIZE_OUT` for stage out jobs. For example, a stage in trigger would look like the following:

```

JOBCFG[dsin]
TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_
move_rsync --stagein",Flags=objectxmlstdin:user:attacherror,Sets=^BYTES_
IN.^DATA_SIZE_IN

```

A stage out trigger would look like the following:

```

JOBCFG[dsout]
TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_
move_rsync --stageout",Flags=objectxmlstdin:user:attacherror,Sets=^BYTES_
OUT.^DATA_SIZE_OUT

```

These variables show up as events if you set your **WIKIEVENTS** parameter to *TRUE*.

- g. Create the stage out job, called `dsout` in the example below, by repeating steps 2b - 2f in a new template. In the example below, this template is called `dsout`. Note that the name of this job template must match the name of the data stage out job template referenced in the data staging master template.

```

JOBCFG[ds]      TEMPLATEDEPEND=AFTEROK:dsin  TEMPLATEDEPEND=BEFORE:dsout
SELECT=TRUE

JOBCFG[dsin]    DATASTAGINGSYSJOB=TRUE
JOBCFG[dsin]    GRES=bandwidth:2
JOBCFG[dsin]    NODEACCESSPOLICY=SINGLEJOB
JOBCFG[dsin]    INHERITRES=TRUE
JOBCFG[dsin]    TRIGGER=EType=start, AType=exec, Action="/opt/moab/tools/data-
staging/ds_move_rsync --stagein", Flags=attacherror:objectxmlstdin:user

JOBCFG[dsout]   DATASTAGINGSYSJOB=TRUE
JOBCFG[dsout]   GRES=bandwidth:1
JOBCFG[dsout]   NODEACCESSPOLICY=SINGLEJOB
JOBCFG[dsout]   INHERITRES=TRUE
JOBCFG[dsout]   TRIGGER=EType=start, AType=exec, Action="/opt/moab/tools/data-
staging/ds_move_rsync --stageout", Flags=attacherror:objectxmlstdin:user

```

3. Create the job using `msub`, adding resources and specifying a script as you normally would. Then configure Moab to stage the data for it. To do so:

- a. If the compute job does not use all of the node's processors, Moab could schedule another job on the node at the same time. If you did not set **NODEACCESSPOLICY** to **SINGLEJOB** in your `moab.cfg`, set the policy for this job by adding `-l naccesspolicy=singlejob` to your `msub` command.

```
> msub -l naccesspolicy=singlejob... <jobScript>
```

- b. At the end of the command, use the `--stagein/--stageout` option and/or `--stageinfile/--stageoutfile` option.
  - The `--stagein/--stageout` option lets you specify a single file or directory to stage in or out. You must set the option equal to `<source>%<destination>`, where `<source>` and `<destination>` are both `[<user>@]<host>:/<path>/[<fileName>]`. See [Staging a file or directory on page 299](#) for format and details.

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

If you do not know the host where the job will run but want the data staged to the same location, you can use the `$JOBHOST` variable in place of a host.

```
> msub --stagein=annasmith@labs:/patient-022678/%$JOBHOST:/davidharris/research/patientrecords <jobScript>
```

*Moab copies the /patient-022678 directory from the hospital's labs server to the node where the job will run prior to job start.*

- The `--stageinfile/--stageoutfile` option lets you specify a file that contains the file and directory name(s) to stage in or out. You must set the option equal to `<path>/<fileName>%` of the file. The file must contain at least one line with this format: `<source>%<destination>`, where `<source>` and `<destination>` are both `[<user>@]<host>:/<path>/[<fileName>]`. See [Staging multiple files or directories on page 300](#) for more information.

**i** If the destination partition is down or does not have configured resources, the data staging workflow submission will fail.

```
> msub --stageinfile=/davidharris/research/recordlist <jobScript>
```

*Moab copies all files specified in the /davidharris/research/recordlist file to the host where the job will run prior to job start.*

```
/davidharris/research/recordlist:
```

```
annasmith@labs:/patient-
022678/tests/blood02282014%$JOBHOST:/davidharris/research/patientrecords/blood
02282014
annasmith@labs:/patient-
022678/visits/stats02032014%$JOBHOST:/davidharris/research/patientrecords/stat
s02032014
annasmith@labs:/patient-
022678/visits/stats02142014%$JOBHOST:/davidharris/research/patientrecords/stat
s02142014
annasmith@labs:/patient-
022678/visits/stats02282014%$JOBHOST:/davidharris/research/patientrecords/stat
s02282014
annasmith@labs:/patient-
022678/visits/stats03032014%$JOBHOST:/davidharris/research/patientrecords/stat
s03032014
annasmith@labs:/patient-
022678/visits/stats03142014%$JOBHOST:/davidharris/research/patientrecords/stat
s03142014
annasmith@labs:/patient-
022678/visits/stats03282014%$JOBHOST:/davidharris/research/patientrecords/stat
s03282014
```

*Moab copies the seven patient record files from the hospital's labs server to the host where the job will run prior to job start.*

- c. The `--stageinsize/--stageoutsize` option lets you specify the estimated size of the files and/or directories to help Moab more quickly and accurately calculate the amount of time it will take to stage the data and therefore schedule your job correctly. If you used the `$JOBHOST` variable to stage in, then setting `--stageinsize` is required. `--stageoutsize` is always required for staging data out. If you provide an integer, Moab will assume the number is in megabytes. To change the unit, add another suffix. See [Stage in or out file size on page 301](#) for more information.

```
> msub --stageinfile=/davidharris/research/recordlist --stageinsize=100
<jobScript>
```

*Moab copies the /davidharris/research/recordlist file, which is approximately 100 megabytes, from the biology node to the host where the job will run prior to job start.*

4. To see the status, errors, and other details associated with your data staging job, run `checkjob -v`. See "[checkjob](#)" for details.

**i** Your `checkjob` output may include a warning that says "req 1 RM (internal) does not match job destination RM". You can safely ignore this message.

## Related topics

- [About data staging on page 901](#)
- [Configuring data staging on page 905](#)
- [Configuring data staging with advanced options on page 920](#)
- [Staging data to or from a compute node on page 916](#)
- [Sample user job script on page 922](#)

## Configuring data staging with advanced options

### Using a different default template name

When you submit a data staging job, a data staging job template is attached to the job automatically. In the reference script configuration, the default template name is `ds`. This is the template that will be attached to the compute job by the client `msub` filter.

If you would like to change the name of the default template that is automatically attached, you should change the value of `DEFAULT_TEMPLATE` in the `ds_config.py` file installed on all client submit hosts. This name must match the master data staging template name specified in the Moab configuration file.

#### To configure the `DEFAULT_TEMPLATE` variable

1. Open the `ds_config.py` file for modification. It is located in `/opt/moab/tools/data-staging/` by default.

```
[moab]$ vi /opt/moab/tools/data-staging/ds_config.py
```

2. Locate the `DEFAULT_PARTITION` parameter.

```
...
DEFAULT_TEMPLATE = "ds"
...
```

3. Replace the template name with the one specified in the Moab configuration file.

```
ds_config.py
...
DEFAULT_TEMPLATE = "datastaging"
...

moab.cfg
...
JOBCFG[datastaging] TEMPLATEDEPEND=...
```

4. Make these changes on all client submit hosts.

### Supporting multiple file transfer script utilities in a grid on a per-partition basis

If you want a different transfer script to run based on which partition the job is submitted to, you can configure a multiplexer script that will switch execution to various other scripts based on the partition.

## To support multiple file transfer script utilities in a grid on a per-partition basis

1. Configure the trigger in your job templates in `moab.cfg` to run `ds_move_multiplex` instead of `ds_move_rsync` or `ds_move_scp`.
2. Configure the `PARTITION_TO_SCRIPT` variable in `ds_config.py` to provide a mapping from each partition to the desired script to run.
  - a. Open the `ds_config.py` file for modification. It is located in `/opt/moab/tools/data-staging/` by default.

```
[moab]$ vi /opt/moab/tools/data-staging/ds_config.py
```

- b. Locate the `PARTITION_TO_SCRIPT` parameter.

```
...
PARTITION_TO_SCRIPT =
{"partition_1_name": "/opt/moab/tools/data-staging/ds_move_rsync",
 "partition_2_name": "/opt/moab/tools/data-staging/ds_move_scp",
 "partition_3_name": "/opt/moab/tools/data-staging/ds_move_rsync"}
...
```

- c. Replace the `partition_*_names` with partitions that exist in your configuration. After each partition, specify the script that you want to execute for that partition.

## Receiving notification at the completion of the data staging job

If you want explicit notification in case of failure of the stage out job, add an additional trigger to the `dsout` job template which will send email notification to the job's submitter. For more information, see [Using a trigger to send email on page 752](#).

```
JOBCFG[dsout]  DATASTAGINGSYSJOB=TRUE
JOBCFG[dsout]  GRES=bandwidth:1
JOBCFG[dsout]  FLAGS=GRESONLY
JOBCFG[dsout]  TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_
move_rsync --stageout",Flags=attacherror:objectxmlstdin:user
JOBCFG[dsout]  TRIGGER=EType=fail,AType=mail,Action="Your (stageout) data staging job
$OID failed."
```

The first trigger listed in the template configuration should be the `exec` trigger. Add the email trigger and any other triggers after the `exec` trigger. You can modify the email trigger to run at completion rather than at failure. You can also add this type of trigger to stage in jobs.

## Adding a non-default template via `msub`

You can have multiple data staging template workflows defined in the `moab.cfg`. The submit filter is configured to add only one of them by default. If you wish to use one of the other available templates, you can do so by using the `-l template=TEMPLATENAME` option in the `msub` command:

Given the following `moab.cfg`:

```
#Default data staging template:
JOBCFG[ds]      TEMPLATEDEPEND=AFTEROK:dsin TEMPLATEDEPEND=BEFORE:dsout SELECT=TRUE
JOBCFG[dsin]   DATASTAGINGSYSJOB=TRUE
JOBCFG[dsin]   GRES=bandwidth:2
JOBCFG[dsin]   FLAGS=GRESONLY
JOBCFG[dsin]   TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_
move_rsync --stagein",Flags=attacherror:objectxmlstdin:user

JOBCFG[dsout]  DATASTAGINGSYSJOB=TRUE
JOBCFG[dsout]  GRES=bandwidth:1
JOBCFG[dsout]  FLAGS=GRESONLY
JOBCFG[dsout]  TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-staging/ds_
move_rsync --stageout",Flags=attacherror:objectxmlstdin:user

#experimental data staging template:
JOBCFG[dscustom] TEMPLATEDEPEND=AFTEROK:dscustomin
TEMPLATEDPEND=BEFORE:dscustomout SELECT=TRUE
JOBCFG[dscustomin] DATASTAGINGSYSJOB=TRUE
JOBCFG[dscustomin] GRES=bandwidth:2
JOBCFG[dscustomin] FLAGS=GRESONLY
JOBCFG[dscustomin] TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_custom --stagein",Flags=attacherror:objectxmlstdin:user

JOBCFG[dscustomout] DATASTAGINGSYSJOB=TRUE
JOBCFG[dscustomout] GRES=bandwidth:1
JOBCFG[dscustomout] FLAGS=GRESONLY
JOBCFG[dscustomout] TRIGGER=EType=start,AType=exec,Action="/opt/moab/tools/data-
staging/ds_move_custom --stageout",Flags=attacherror:objectxmlstdin:user
```

The user could submit a job using the custom data staging template with the following command:

```
[moab]$ msub -l template=dscustom ...
```

## Using msub to return all the job IDs in the workflow at submission time

By default, `msub` will print the job ID to stdout at the time of submission. If you would like to have `msub` print all of the jobs that are created as part of the data staging workflow template, you can use the `msub --workflowjobids` option to show all the job IDs at submission time:

```
$ echo sleep 60 | msub -l walltime=15 --workflowjobids
MoabA.3.dsin MoabA.3 MoabA.3.dsout
```

This could be useful if you are writing scripts to do your own workflows and you need to programmatically capture the data stage out job name for use in your workflow.

Related topics

- [Configuring data staging on page 905](#)

## References

### Sample user job script

The code below is an example of a job script that a user might use to run a data staging job.

```
#!/bin/bash
#
# Sample data staging job script
#
# stage in directives
#MSUB --stageinsize=1MB
#MSUB --stagein=davidharris@source-server:/tmp/filein.tostage%davidharris@destination-
server:/tmp/filein.staged
#
# stage out directives
#MSUB --stageoutsize=10GB
#MSUB --stageout=davidharris@destination-
server:/tmp/fileout.tostage%davidharris@source-server:/tmp/fileout.staged

# run executable on the destination host using staged data
$HOME/bin/my_compute_executable < /tmp/filein.staged > /tmp/fileout.tostage
```

Related topics

- [About data staging on page 901](#)
- [Staging data on page 299](#)

# Appendices

## Appendix A: Moab Parameters

See the [Parameters Overview](#) in the Moab Admin Manual for further information about specifying parameters.

Index: [A](#) [B](#) [C](#) [D](#) [E](#) [F](#) [G](#) [H](#) [I](#) [J](#) [K](#) [L](#) [M](#) [N](#) [O](#) [P](#) [Q](#) [R](#) [S](#) [T](#) [U](#) [V](#) [W](#) [X](#) [Y](#) [Z](#)

ACCOUNTCFG[<ACCOUNTID>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">General Credential Flags</a> , <b>CHARGERATE</b> , <a href="#">PRIORITY</a> , <a href="#">ENABLEPROFILING</a> , <a href="#">MEMBERULIST</a> , <b>PLIST</b> , <b>QDEF</b> , <b>QLIST</b> , <a href="#">usage limit</a> , or a <a href="#">fairness usage limit</a> specification ( <a href="#">FSCAP</a> , <a href="#">FSTARGET</a> , and <a href="#">FSWEIGHT</a> ).
<b>Default</b>	---
<b>Description</b>	Specifies account specific attributes. See the <a href="#">account</a> overview for general information and the <a href="#">job flag overview</a> for a description of legal flag values.
<b>Example</b>	<pre>ACCOUNTCFG[projectX] MAXJOB=50 QDEF=highprio</pre> <p><i>Up to 50 jobs submitted under the account ID projectX will be allowed to execute simultaneously and will be assigned the QOS highprio by default.</i></p>

ACCOUNTINGINTERFACEURL	
<b>Format</b>	<URL> where protocol can be one of <b>exec</b> or <b>file</b>
<b>Default</b>	---
<b>Description</b>	Specifies the interface to use for real-time export of Moab accounting/auditing information. See <a href="#">Exporting Events in Real-Time</a> for more information.
<b>Example</b>	<pre>ACCOUNTINGINTERFACEURL exec:/// \$TOOLS DIR/dumpacc.pl</pre>

ACCOUNTWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the priority weight to be applied to the specified account priority. See <a href="#">Credential (CRED) Factor</a> .
<b>Example</b>	<pre>ACCOUNTWEIGHT 100</pre>

ADMIN <sub>1</sub> , ADMIN <sub>2</sub> , ADMIN <sub>3</sub>	
<b>Format</b>	Space-delimited list of user names
<b>Default</b>	<i>root</i>
<b>Description</b>	Deprecated. Use <a href="#">ADMINCFG</a> . Users listed under the parameter <b>ADMIN<sub>i</sub></b> are allowed to perform any scheduling function. They have full control over the scheduler and access to all data. The first user listed in the <b>ADMIN<sub>i</sub></b> user list is considered to be the 'primary admin' and is the ID under which Moab must be started and run. Valid values include user names or the keyword 'ALL'. Again, these parameters are deprecated; use <a href="#">ADMINCFG</a> .
<b>Example</b>	<pre>ADMIN1 moabuser steve scott jenny</pre> <i>All users listed have full access to Moab control commands and Moab data. Moab must be started by and run under the <b>moabuser</b> user id since <b>moabuser</b> is the primary admin.</i>

ADMINCFG[X]	
<b>Format</b>	One or more <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <b>ENABLEPROXY</b> , <b>USERS</b> , <b>GROUPS</b> , <b>SERVICES</b> , or <b>NAME</b>
<b>Default</b>	---
<b>Description</b>	Allows a site to configure which services and users belong to a particular level of administration. <b>Note:</b> The first user listed in the <b>ADMINCFG[1]</b> users list is considered to be the primary admin. The option <b>USERS=ALL</b> is allowed. The groups list adds the groups' users as if they were listed individually as <b>USERS</b> . To prevent Moab from assigning a primary user from the first group listed, you must specify a primary user first using the <b>USERS</b> attribute, then list the desired groups.
<b>Example</b>	<pre>ADMINCFG[1] USERS=root, john ADMINCFG[1] GROUPS=admin ADMINCFG[1] SERVICES=ALL ADMINCFG[1] NAME=batchadmin ADMINCFG[3] USERS=bob, carol, smoore ADMINCFG[3] GROUPS=science, math ADMINCFG[3] SERVICES=mjobctl, mcredctl, runjob ADMINCFG[3] NAME=helpdesk</pre> <p><i>Members of the <b>batchadmin</b> admin role and members of the <b>admin</b> group are allowed to run all commands. Members of the <b>helpdesk</b> role and <b>science</b> and <b>math</b> groups are allowed to run <b>mjobctl</b>. They are also able to view and modify credential objects (i.e. users, groups, accounts, etc.) See the <a href="#">security overview</a> for more details.</i></p> <pre>ADMINCFG[4] USERS=ALL SERVICES=checknode</pre> <p><i>All users can execute <b>mdiag -n</b> or <b>checknode</b> to get information on any node.</i></p>

AGGREGATENODEACTIONS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	<p>Consolidates queued node actions into as few actions as possible to reduce communication burden with resource manager. Node actions are queued until the <a href="#">AGGREGATENODEACTIONSTIME</a> setting.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> This may delay some node actions. When reprovisioning, the system job may expire before the provision action occurs; while the action will still occur, the job will not show it.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-bottom: 5px;"> <pre>AGGREGATENODEACTIONS TRUE</pre> </div> <div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-left: 20px;"> <p><i>Queues node actions together when possible.</i></p> </div>

AGGREGATENODEACTIONSTIME	
<b>Format</b>	<SECONDS>
<b>Default</b>	60
<b>Description</b>	<p>The delay time for the <a href="#">AGGREGATENODEACTIONS</a> parameter to aggregate requests before sending job batches.</p>
<b>Example</b>	<div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-bottom: 5px;"> <pre>AGGREGATENODEACTIONSTIME 120</pre> </div> <div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-left: 20px;"> <p><i>Sets the <b>AGGREGATENODEACTIONS</b> delay to two minutes.</i></p> </div>

ALLOWMULTIREQNODEUSE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	By default Moab does not allow different requirements on the same job to occupy the same node. For example, if a job is submitted with <code>nodes=2:ppn=8+4:fast:ppn=16</code> , it's possible that some of the tasks requested could overlap onto the same node. This parameter instructs Moab to allow overlapping the same node, or not. This parameter also applies to the various <code>-w</code> clauses of an <a href="#">mshow -a on page 272</a> command.
<b>Example</b>	<pre>ALLOWMULTIREQNODEUSE TRUE</pre>

ALLOWROOTJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether batch jobs from the root user (UID=0) are allowed to be executed. <b>Note:</b> The resource manager must also support root jobs.
<b>Example</b>	<pre>ALLOWROOTJOBS TRUE</pre> <i>Jobs from the root user can execute.</i>

ALLOWVMMIGRATION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Enables Moab to migrate VMs.
<b>Example</b>	<pre>ALLOWVMMIGRATION TRUE</pre>

ALWAYSEVALUATEALLJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	When scheduling priority jobs, Moab stops scheduling when it encounters the first job that cannot run and cannot get a reservation. <b>ALWAYSEVALUATEALLJOBS</b> directs Moab to continue scheduling until all priority jobs (jobs that do not violate any limits) are evaluated.
<b>Example</b>	<pre>ALWAYSEVALUATEALLJOBS TRUE</pre>

AMCFG	
<b>Format</b>	One or more key-value pairs as described in the Allocation Manager Configuration Overview.
<b>Default</b>	---
<b>Description</b>	Specifies the interface and policy configuration for the scheduler-allocation manager interface. Described in detail in the Allocation Manager Configuration Overview.
<b>Example</b>	<pre>AMCFG[mam] SERVER=mam://master.ufl.edu STARTFAILUREACTION=HOLD TIMEOUT=15</pre>

APPLICATIONLIST	
<b>Format</b>	Space-delimited list of generic resources.
<b>Default</b>	---
<b>Description</b>	Specifies which generic resources represent actual applications on the cluster/grid. See <a href="#">Managing Consumable Generic Resources</a> for more information.
<b>Example</b>	<pre>NODECFG[node01] GRES=calclab:1,powerhouse:1 RCSOFTWARE=calclab:1,powerhouse:1 NODECFG[node02] GRES=calclab:1,powerhouse:1 RCSOFTWARE=calclab:1,powerhouse:1 APPLICATIONLIST calclab,powerhouse</pre> <p><i>The generic resources <b>calclab</b> and <b>powerhouse</b> will now be recognized and treated as application software.</i></p>

ARRAYJOBPARLOCK	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , all sub jobs of an array are locked to a single partition. The default behavior when scheduling array sub jobs is to span the jobs across partitions when possible. The <a href="#">ARRAYJOBPARLOCK</a> job flag can be used to specify partition locking at submit time. The <a href="#">ARRAYJOBPARSPAN</a> job flag overrides this parameter.
<b>Example</b>	<pre>ARRAYJOBPARLOCK TRUE</pre>

ASSIGNVLANFEATURES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	When set to <i>TRUE</i> , this forces all VMs to be contained in VLANs.
<b>Example</b>	<pre>ASSIGNVLANFEATURES TRUE</pre>

ATTRATTRWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>0</i>
<b>Description</b>	Specifies the priority weight to be applied to jobs with the specified job attribute. See <a href="#">Attribute (ATTR) Factor</a> .
<b>Example</b>	<pre>ATTRATTRWEIGHT 100</pre>

ATTRGRESWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight to be applied to jobs requesting the specified <a href="#">generic resource</a> . See <a href="#">Attribute (ATTR) Factor</a> .
<b>Example</b>	<pre>ATTRGRESWEIGHT 200</pre>

ATTRSTATEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight to be applied to jobs with the specified job state. See <a href="#">Attribute (ATTR) Factor</a> .
<b>Example</b>	<pre>ATTRSTATEWEIGHT 200</pre>

ATTRWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the priority component weight to be applied to the ATTR subcomponents. See <a href="#">Attribute (ATTR) Factor</a> .
<b>Example</b>	<pre>ATTRWEIGHT 2 ATTRSTATEWEIGHT 200</pre>

BACKFILLDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (no limit)
<b>Description</b>	Specifies the number of idle jobs to evaluate for backfill. The backfill algorithm will evaluate the top <X> priority jobs for scheduling. By default, all jobs are evaluated.
<b>Example</b>	<pre>BACKFILLDEPTH 128</pre> <p><i>Evaluate only the top 128 highest priority idle jobs for consideration for backfill.</i></p>

BACKFILLPOLICY	
<b>Format</b>	One of <i>FIRSTFIT</i> or <i>NONE</i>
<b>Default</b>	<i>FIRSTFIT</i>
<b>Description</b>	Specifies which backfill <a href="#">algorithm</a> will be used. See <a href="#">Configuring Backfill</a> for more information.
<b>Example</b>	<pre>BACKFILLPOLICY NONE</pre>

BFCHUNKDURATION	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0 (chunking disabled)
<b>Description</b>	Specifies the duration during which freed resources will be aggregated for use by larger jobs. Used in conjunction with <a href="#">BFCHUNKSIZE on page 932</a> . See <a href="#">Configuring Backfill</a> for more information.
<b>Example</b>	<pre>BFCHUNKDURATION 00:05:00 BFCHUNKSIZE 4</pre> <p><i>Aggregate backfillable resources for up to 5 minutes, making resources available only to jobs of size 4 or larger.</i></p>

BFCHUNKSIZE	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (chunking disabled)
<b>Description</b>	Specifies the minimum job size which can utilize chunked resources. Used in conjunction with <a href="#">BFCHUNKDURATION</a> on page 931. See <a href="#">Configuring Backfill</a> for more information.
<b>Example</b>	<pre>BFCHUNKDURATION 00:05:00 BFCHUNKSIZE      4</pre> <p><i>Aggregate backfillable resources for up to 5 minutes, making resources available only to jobs of size 4 or larger.</i></p>

BFMINVIRTUALWALLTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	Specifies the minimum job wallclock time for virtual scaling (optimistic-like backfilling.) Any job with a wallclock time less than this setting will <i>not</i> be virtually scaled. The value specified relates to a job's original walltime and not its virtually-scaled walltime.
<b>Example</b>	<pre>BFMINVIRTUALWALLTIME 00:01:30</pre>

BFPRIORITYPOLICY	
<b>Format</b>	One of <i>RANDOM</i> , <i>DURATION</i> , or <i>HWDURATION</i>
<b>Default</b>	---
<b>Description</b>	Specifies policy to use when prioritizing backfill jobs for <a href="#">preemption</a>
<b>Example</b>	<pre>BFPRIORITYPOLICY DURATION</pre> <p><i>Use length of job in determining which backfill job to preempt.</i></p>

BFVIRTUALWALLTIMECONFLICTPOLICY	
<b>Format</b>	One of the following: <i>PREEMPT</i>
<b>Default</b>	---
<b>Description</b>	Specifies how to handle scheduling conflicts when a virtually scaled job "expands" to its original wallclock time. This occurs when the job is within one scheduling iteration - <a href="#">RMPOLLINTERVAL on page 1036</a> - of its virtually scaled wallclock time expiring.
<b>Example</b>	<pre>BFVIRTUALWALLTIMECONFLICTPOLICY PREEMPT</pre>

BFVIRTUALWALLTIMESCALINGFACTOR	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (virtual scaling disabled)
<b>Description</b>	Specifies the factor by which eligible jobs' wallclock time is virtually scaled (optimistic-like backfilling).
<b>Example</b>	<pre>BFVIRTUALWALLTIMESCALINGFACTOR .4</pre>

BYPASSCAP	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the max weighted value allowed from the bypass count subfactor when determining a job's priority (see <a href="#">Priority Factors</a> for more information).
<b>Example</b>	<pre>BYPASSWEIGHT 5000 BYPASSCAP 30000</pre>

BYPASSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight to be applied to a job's backfill bypass count when determining a job's priority (see <a href="#">Priority Factors</a> for more information).
<b>Example</b>	<pre>BYPASSWEIGHT 5000</pre>

CHECKPOINTDIR	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the directory for temporary job checkpoint files (usually of the form <code>jobid.cp</code> ). This is <i>not</i> the directory for Moab's checkpoint file ( <code>.moab.ck</code> ).
<b>Example</b>	<pre>CHECKPOINTDIR /tmp/moabcheckpoint</pre>

CHECKPOINTEXPIRATIONTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS or <i>UNLIMITED</i>
<b>Default</b>	<i>3,000,000 seconds</i>
<b>Description</b>	Specifies how 'stale' checkpoint data can be before it is ignored and purged.
<b>Example</b>	<pre>CHECKPOINTEXPIRATIONTIME 1:00:00:00</pre> <p><i>Expire checkpoint data which has been stale for over 1 day.</i></p>

CHECKPOINTFILE	
<b>Format</b>	<STRING>
<b>Default</b>	<i>moab.ck</i>
<b>Description</b>	Name (absolute or relative) of the Moab checkpoint file.
<b>Example</b>	<pre>CHECKPOINTFILE /var/adm/moab/moab.ck</pre> <p><i>Maintain the Moab checkpoint file in the file specified.</i></p>

CHECKPOINTINTERVAL	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	<i>00:05:00</i>
<b>Description</b>	<p>Time between automatic Moab checkpoints.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p><b>i</b> If <b>RMPOLLINTERVAL</b> on page 1036 does not specify both a minimum and maximum poll time, Moab will ignore <b>CHECKPOINTINTERVAL</b> and checkpoint every iteration.</p> </div>
<b>Example</b>	<pre>CHECKPOINTINTERVAL 00:15:00</pre> <p><i>Moab should checkpoint state information every 15 minutes.</i></p>

CHECKPOINTWITHDATABASE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , Moab stores checkpoint information to a database rather than to the <code>.moab.ck</code> flat text file.
<b>Example</b>	<pre>CHECKPOINTWITHDATABASE TRUE</pre>

CHECKSUSPENDEDJOBPRIORITY	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	Prevents Moab from starting a job on any node containing a suspended job of higher priority.
<b>Example</b>	<pre>CHECKSUSPENDEDJOBPRIORITY FALSE</pre>

CHILDSTDERRCHECK	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , child processes Moab executes are considered failed if their standard error stream contains the text "ERROR".
<b>Example</b>	<pre>CHILDSTDERRCHECK TRUE</pre>

CLASSCFG[<CLASSID>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">General Credential Flags</a> , <a href="#">DEFAULT.ATTR</a> , <a href="#">DEFAULT.DISK</a> , <a href="#">DEFAULT.FEATURES</a> , <a href="#">DEFAULT.GRES</a> , <a href="#">DEFAULT.MEM</a> , <a href="#">DEFAULT.NODE</a> , <a href="#">DEFAULT.NODESET</a> , <a href="#">DEFAULT.PROC</a> , <a href="#">ENABLEPROFILING</a> , <a href="#">EXCL.FEATURES</a> , <a href="#">EXCLUDEUSERLIST</a> , <a href="#">HOSTLIST</a> , <a href="#">JOBPILOG</a> , <a href="#">JOBPROLOG</a> , <a href="#">MAXPROCERNODE</a> , <a href="#">MAX.NODE</a> , <a href="#">MAX.PROC</a> , <a href="#">MAX.WCLIMIT</a> , <a href="#">MIN.NODE</a> , <a href="#">MIN.PROC</a> , <a href="#">MIN.TPN</a> , <a href="#">MIN.WCLIMIT</a> , <a href="#">PARTITION</a> , <a href="#">PRIORITY</a> , <a href="#">PRIORITYF</a> , <a href="#">QDEF</a> , <a href="#">QLIST</a> , <a href="#">REQ.FEATURES</a> , <a href="#">REQUIREDACCOUNTLIST</a> , <a href="#">REQUIREDUSERLIST</a> , <a href="#">RESFAILPOLICY</a> , <a href="#">SYSPRIO</a> , <a href="#">WCOVERRUN</a> , <a href="#">usage limit</a> , or <a href="#">fairshare usage limit</a> specification.
<b>Default</b>	---
<b>Description</b>	Specifies class specific attributes (see <a href="#">Credential Overview</a> for details).
<b>Example</b>	<pre>CLASSCFG[batch] MAXJOB=50 QDEF=highprio</pre> <i>Up to 50 jobs submitted to the class batch will be allowed to execute simultaneously and will be assigned the QOS highprio by default.</i>

CLASSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the weight to be applied to the class priority of each job (see <a href="#">Credential (CRED) Factor</a> and <a href="#">credential priority</a> ).
<b>Example</b>	<pre>CLASSWEIGHT 10</pre>

CLIENTCFG[<X>]	
<b>Format</b>	One or more of <ATTR>-<VALUE> pairs where <X> indicates the specified peer and <ATTR> is one of the following: <b>AUTH</b> , <b>AUTHCMD</b> , <b>AUTHTYPE</b> , <b>HOST</b> , <b>KEY</b> , or <b>DEFAULTSUBMITPARTITION</b> .
<b>Default</b>	---
<b>Description</b>	Specifies the shared secret key and authentication method which Moab will use to communicate with the named peer daemon. See <a href="#">Security Overview</a> for more information. <b>Note:</b> The <b>AUTHTYPE</b> and <b>KEY</b> attributes of this parameter may only be specified in the <code>moab-private.cfg</code> config file.
<b>Example</b>	<pre>CLIENTCFG[silverB] KEY=apple7 AUTH=admin1</pre> <p><i>Moab will use the session key <b>apple7</b> for peer authentication and for encrypting and decrypting messages sent from <code>silverB</code>. Also, client connections from this interface will be authorized at an <b>admin 1</b> level.</i></p>

CLIENTMAXCONNECTIONS	
<b>Format</b>	<INTEGER>
<b>Default</b>	128
<b>Description</b>	Changes the maximum number of connections that can simultaneously connect to Moab. The value can be increased during runtime, but it cannot be decreased. The value cannot be reduced below the default value of 128.
<b>Example</b>	<pre>CLIENTMAXCONNECTIONS 256</pre> <p><i>Doubles the maximum number of connections.</i></p>

CLIENTMAXPRIMARYRETRY	
<b>Format</b>	<INTEGER> or <i>INFINITY</i>
<b>Default</b>	1
<b>Description</b>	Specifies how many times the client command will attempt to retry its connection to the primary server if Moab is not available.
<b>Example</b>	<pre>CLIENTMAXPRIMARYRETRY 5 CLIENTMAXPRIMARYRETRYTIMEOUT 1000</pre> <p><i>The client command will attempt to retry its connection to the primary server 5 times with 1 second intervals before giving up. <b>Note:</b> If <i>INFINITY</i> is specified, Moab will attempt 2,140,000,000 times.</i></p>

CLIENTMAXPRIMARYRETRYTIMEOUT	
<b>Format</b>	<INTEGER> (milliseconds)
<b>Default</b>	2000
<b>Description</b>	Specifies how much time to wait until the client command will attempt to retry its connection to the primary server if Moab is not available.
<b>Example</b>	<pre>CLIENTMAXPRIMARYRETRY 3 CLIENTMAXPRIMARYRETRYTIMEOUT 500</pre> <p><i>The client command will attempt to retry its connection to the primary server 3 times with .5 second intervals before giving up.</i></p>

CLIENTTIMEOUT	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>00:00:30</code>
<b>Description</b>	Time which Moab client commands will wait for a response from the Moab server. See <a href="#">Client Configuration</a> for more information. <b>Note:</b> May also be specified as an environment variable.
<b>Example</b>	<pre>CLIENTTIMEOUT 00:15:00</pre> <p><i>Moab clients will wait up to 15 minutes for a response from the server before timing out.</i></p>

CREDDISCOVERY	
<b>Format</b>	<code>TRUE</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	Specifies that Moab should create otherwise unknown credentials when it discovers them in the statistics files.
<b>Example</b>	<pre>CREDDISCOVERY TRUE</pre>

CREDWEIGHT	
<b>Format</b>	<code>&lt;INTEGER&gt;</code>
<b>Default</b>	<code>1</code>
<b>Description</b>	Specifies the credential component weight associated with the <a href="#">credential priority</a> . See <a href="#">Credential (CRED) Factor</a> for more information.
<b>Example</b>	<pre>CREDWEIGHT 2</pre>

DATASTAGEHOLDTYPE	
<b>Format</b>	Any <a href="#">Job Hold</a> type
<b>Default</b>	<i>DEFER</i>
<b>Description</b>	Specifies what to do if a job's data staging operations fail.
<b>Example</b>	<code>DATASTAGEHOLDTYPE BATCH</code>

DEADLINEPOLICY	
<b>Format</b>	One of <i>CANCEL</i> , <i>HOLD</i> , <i>IGNORE</i> , or <i>RETRY</i>
<b>Default</b>	<i>NONE</i>
<b>Description</b>	Specifies what to do when a requested deadline cannot be reached (see <a href="#">Job Deadlines</a> ).
<b>Example</b>	<code>DEADLINEPOLICY IGNORE</code>

DEFAULTCLASSLIST	
<b>Format</b>	Space-delimited list of one or more <i>&lt;STRING&gt;</i> s.
<b>Default</b>	---
<b>Description</b>	Specifies the default classes supported on each node for RM systems which do not provide this information.
<b>Example</b>	<code>DEFAULTCLASSLIST serial parallel</code>

DEFAULTSUBMITPARTITION	
<b>Format</b>	See parameter <a href="#">CLIENTCFG[]</a> .
<b>Default</b>	---
<b>Description</b>	If a user submits a job using <a href="#">msub</a> which does not specify host, feature, or partition constraints, then the <a href="#">msub</a> client will insert the specified default submit partition into the newly submitted job as a hard requirement.
<b>Example</b>	<pre>CLIENTCFG[DEFAULT] DEFAULTSUBMITPARTITION=partition1</pre>

DEFERCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	24
<b>Description</b>	Specifies the number of times a job can be deferred before it will be placed in batch hold.
<b>Example</b>	<pre>DEFERCOUNT 12</pre>

DEFERSTARTCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the number of times a job will be allowed to fail in its start attempts before being deferred. <b>JOBRETRYTIME</b> overrides <b>DEFERSTARTCOUNT</b> ; <b>DEFERSTARTCOUNT</b> only begins when the <b>JOBRETRYTIME</b> window elapses. <b>Note:</b> A job's startcount will increase each time a start request is made to the resource manager regardless of whether or not this request succeeded. This means start count increases if job starts fail or if jobs are started and then rejected by the resource manager. (For related information, see <a href="#">Reservation Policies</a> , <a href="#">DEFERTIME</a> , <a href="#">RESERVATIONRETRYTIME</a> , <a href="#">NODEFAILURERESERVETIME</a> , <a href="#">JOBRETRYTIME</a> , and <a href="#">GUARANTEEDPREEMPTION</a> .)
<b>Example</b>	<pre>DEFERSTARTCOUNT 3</pre>

DEFERTIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>1:00:00</code>
<b>Description</b>	Specifies the amount of time a job will be held in the deferred state before being released back to the Idle job queue. <b>Note:</b> A job's defer time will be restarted if Moab is restarted. (For related information, see <a href="#">Reservation Policies</a> , <a href="#">DEFERSTARTCOUNT</a> , <a href="#">RESERVATIONRETRYTIME</a> , <a href="#">NODEFAILURERESERVETIME</a> , <a href="#">JOBRETRYTIME</a> , and <a href="#">GUARANTEEDPREEMPTION</a> .)
<b>Example</b>	<pre>DEFERTIME 0:05:00</pre>

DELETESTAGEOUTFILES	
<b>Format</b>	<code>&lt;BOOLEAN&gt;</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	Specifies whether the scheduler should delete explicitly specified stageout files after they are successfully staged. By default, such files are not deleted but are left on the nodes where the job ran.
<b>Example</b>	<pre>DELETESTAGEOUTFILES TRUE Example of an explicit stageout request msub x=MSTAGEOUT:ssh://source_node/tmp/file,file:///results_folder job.cmd</pre> <p><i>With this parameter set to <b>TRUE</b>, /tmp/file on source_node is deleted after it is copied to the specified destination ( file:///results_folder). If the parameter is not set, or if it is set to <b>FALSE</b>, /tmp/file remains on source_node after the job terminates.</i></p>

DEPENDFAILUREPOLICY	
<b>Format</b>	<i>HOLD</i> or <i>CANCEL</i>
<b>Default</b>	<i>HOLD</i>
<b>Description</b>	Specifies what happens to a job if its dependencies cannot be fulfilled; that is, what happens when a job depends on another job to complete successfully but the other job fails.
<b>Example</b>	<pre>DEPENDFAILUREPOLICY CANCEL</pre> <p><i>If job A is submitted with depend=afterok:B and job B fails, job A is canceled.</i></p>

DIRECTORYSERVER	
<b>Format</b>	<HOST>[:<PORT>]
<b>Default</b>	---
<b>Description</b>	Specifies the interface for the directory server.
<b>Example</b>	<pre>DIRECTORYSERVER calli3.icluster.org:4702</pre>

DISABLEEXCHLIST	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If the resource manager rejects a job and the value is set to <i>TRUE</i> , then the node is not added to the job's exclude host list.
<b>Example</b>	<pre>DISABLEEXCHLIST TRUE</pre>

DISABLEINTERACTIVEJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>Disallows interactive jobs submitted with <a href="#">msub -I</a>.</p> <p><b>Note:</b> It is possible for users to submit interactive jobs directly to a resource manager, which can bypass the <b>DISABLEINTERACTIVEJOBS</b> parameter. However, some resource managers (such as TORQUE) will check with Moab before allowing an interactive job.</p>
<b>Example</b>	<pre>DISABLEINTERACTIVEJOBS TRUE</pre>

DISABLEREGEXCACHING	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Turns off regular expression caching. Turning off regular expression caching preserves memory with hostlist reservations and speeds up start time.
<b>Example</b>	<pre>DISABLEREGEXCACHING TRUE</pre>

DISABLEREQUIREDGRESNONE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	When set to <i>TRUE</i> , this causes Moab to reject msub requests that have a gres of "none". <a href="#">ENFORCEGRESACCESS</a> must also be set to TRUE for this feature to work.
<b>Example</b>	<pre>##### moab.cfg ##### ENFORCEGRESACCESS TRUE DISABLEREQUIREDGRESNONE TRUE #####  &gt; msub -A ee -l nodes=1,ttc=5,walltime=600,partition=g02 -l gres=none ERROR: cannot submit job - cannot locate required resource 'none'</pre>

DISABLESAMECREDPREEMPTION	
<b>Format</b>	Comma-delimited list of one or more credentials: <i>ACCT</i> , <i>CLASS</i> , <i>GROUP</i> , <i>QOS</i> , or <i>USER</i>
<b>Default</b>	---
<b>Description</b>	This parameter prevents specified credentials from preempting its own jobs.
<b>Example</b>	<pre>DISABLESAMECREDPREEMPTION QOS,USER</pre>

DISABLESCHEDULING	
<b>Format</b>	< <i>BOOLEAN</i> >
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether or not the scheduler will schedule jobs. If set to <i>TRUE</i> , Moab will continue to update node and job state but will not start, preempt, or otherwise modify jobs. The command <a href="#">mschedctl -r</a> will clear this parameter and resume normal scheduling.
<b>Example</b>	<pre>DISABLESCHEDULING FALSE</pre>

DISABLESLAVEJOBSSUBMIT	
<b>Format</b>	< <i>BOOLEAN</i> >
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	This parameter can be added to the <code>moab.cfg</code> file on a slave Moab server (in a grid configuration) to prevent users from submitting jobs to the master Moab server from the slave Moab server. Some grid configurations allow the user to submit jobs on the slave that are migrated to the master and submitted from the master. Other grid configurations do not allow the jobs to be migrated to the master from the slave, in which case, jobs submitted from the slave remain idle on the slave and never run. This parameter will reject the job submissions on the slave to prevent the submission of jobs that will never run.
<b>Example</b>	<pre>DISABLESLAVEJOBSSUBMIT TRUE example (node04 is a slave and node06 is the master) [test@node04 moab-slurm]\$ echo sleep 100   msub ERROR: cannot submit job from slave</pre>

DISABLETHRESHOLDTRIGGERS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	This makes Moab not fire threshold-based triggers, but will log the intended action to the event logs. Similar to <a href="#">DISABLEVMDECISIONS</a> .
<b>Example</b>	<pre>DISABLETHRESHOLDTRIGGERS TRUE</pre>

DISABLEVMDECISIONS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	This makes Moab not take any automatic decisions with respect to VM's, namely powering on/off nodes and migrating VMs. Intended actions will instead be logged in the event logs. Similar to <a href="#">DISABLETHRESHOLDTRIGGERS</a> .
<b>Example</b>	<pre>DISABLEVMDECISIONS TRUE</pre>

DISKWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight to be applied to the amount of dedicated disk space required per task by a job (in MB).
<b>Example</b>	<pre>RESWEIGHT 10 DISKWEIGHT 100</pre> <p><i>If a job requires 12 tasks and 512 MB per task of dedicated local disk space, Moab will increase the job's priority by <math>10 * 100 * 12 * 512</math></i></p>

DISPLAYFLAGS	
<b>Format</b>	One or more of the following values (space delimited): <i>ACCOUNTCENTRIC</i> , <i>HIDEBLOCKED</i> , <i>HIDEDRAINED</i> , <i>NODECENTRIC</i> , or <i>USEBLOCKING</i>
<b>Default</b>	---
<b>Description</b>	<p>Specifies flags that control how Moab client commands display varied information.</p> <p><i>ACCOUNTCENTRIC</i> will display account information in <a href="#">showq</a>, rather than group information.</p> <p><i>HIDEBLOCKED</i> will prevent <a href="#">showq</a> from listing information about blocked jobs which are not owned by the user if the user is not an admin.</p> <p><i>HIDEDRAINED</i> prevents <a href="#">mdiag -n</a> from displaying nodes and <a href="#">mvmctl -q</a> from displaying VMs in the <a href="#">DRAINED</a> state. An override option of <a href="#">mdiag -n -w nodestate=drained</a> lists only those nodes with a <a href="#">DRAINED</a> state. Similarly, an override option of <a href="#">mvmctl -q -w state=drained</a> lists only those VMs with a <a href="#">DRAINED</a> state.</p> <p><i>NODECENTRIC</i> will display node allocation information instead of processor allocation information in <a href="#">showq</a>.</p> <p><i>USEBLOCKING</i> disables threading for commands that support it; those commands include <a href="#">showq</a>, <a href="#">mdiag -n</a>, and <a href="#">mdiag -j</a>.</p>
<b>Example</b>	<pre>DISPLAYFLAGS NODECENTRIC</pre>

DISPLAYPROXYUSERASUSER	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , Moab shows the proxy users instead of the real user on some queries of system jobs that have proxy users. Commands affected include <a href="#">mjobctl -q diag</a> and <a href="#">checkjob</a> .
<b>Example</b>	<pre>DISPLAYPROXYUSERASUSER TRUE</pre>

DONTCANCELINTERACTIVEHJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , Moab does not cancel interactive jobs that are held.
<b>Example</b>	<code>DONTCANCELINTERACTIVEHJOBS TRUE</code>

DONTENFORCEPEERJOBLIMITS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , only the scheduler that is running the job can cancel the job or enforce other limits.
<b>Example</b>	<code>DONTENFORCEPEERJOBLIMITS TRUE</code>

EMULATIONMODE	
<b>Format</b>	<i>SLURM</i>
<b>Default</b>	---
<b>Description</b>	Specifies whether or not the scheduler will perform the automatic setup of a particular resource manager environment.
<b>Example</b>	<pre>EMULATIONMODE SLURM</pre> <p><i>Moab will perform the automated setup steps as if it were interfacing with a slurm resource manager (automatic QOS creation).</i></p>

ENABLEFAILUREFORPURGEDJOB	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>By default, when a job is purged or removed by the TORQUE resource manager for a walltime violation, the job takes on a state of Completed and a completion code of 0. If <i>TRUE</i>, the job state is set to Removed and has a completion code of 98.</p> <p><b>ENABLEFAILUREFORPURGEDJOB</b> is for the TORQUE resource manager only.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #D9E1F2;"> <p><b>i</b> For <b>ENABLEFAILUREFORPURGEDJOB</b> to return Removed job states, you must reset the TORQUE server attribute <code>keep_completed</code> to 0 in <code>qmgr</code>. See "<a href="#">Queue attributes on page 2626</a>" in the TORQUE Administrator Guide for more information.</p> </div>
<b>Example</b>	<pre>ENABLEFAILUREFORPURGEDJOB TRUE</pre> <p><i>Jobs that are purged or removed by TORQUE are given a state of Removed and a completion code of 98.</i></p>

ENABLEFSVIOLATIONPREEMPTION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If set to <b>TRUE</b> , Moab will allow jobs within the same <a href="#">class/queue</a> to <a href="#">preempt</a> when the preemptee is violating a <a href="#">fairshare</a> target and the preemptor is not.
<b>Example</b>	<pre>ENABLEFSVIOLATIONPREEMPTION TRUE</pre>

ENABLEHIGHTHROUGHPUT	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	<p>Configures Moab so that it will accept <code>msub</code> submissions, start jobs, process triggers, etc., in a manner which minimizes their processing time. The downside is that Moab will return minimal information about these jobs at submit time.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF;"> <p><b>i</b> If <b>ENABLEHIGHTHROUGHPUT</b> is <b>TRUE</b>, you must set <b>NODEALLOCATIONPOLICY</b> on <a href="#">page 999</a> to <b>FIRSTAVAILABLE</b>.</p> </div>
<b>Example</b>	<pre>ENABLEHIGHTHROUGHPUT TRUE</pre> <p><i>Moab can now accept hundreds of jobs per second using msub instead of 20-30.</i></p>

ENABLEJOBARRAYS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	If set to <b>TRUE</b> , job arrays will be enabled .
<b>Example</b>	<pre>ENABLEJOBARRAYS TRUE</pre>

ENABLENEGJOBPRIORITY	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , the scheduler allows job priority value to range from -INFINITY to MMAX_PRIO; otherwise, job priority values are given a lower bound of '1'. For more information, see <a href="#">REJECTNEGPRIOJOBS</a> .
<b>Example</b>	<pre>ENABLENEGJOBPRIORITY TRUE</pre> <p><i>Job priority may range from -INFINITY to MMAX_PRIO.</i></p>

ENABLENODEADDRLOOKUP	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , the scheduler will use the default host name service lookup mechanism (i.e., /etc/hosts, DNS, NIS, etc.) to determine the IP address of the nodes reported by the resource manager. This information is used to correlate information reported by multi-homed hosts.
<b>Example</b>	<pre>ENABLENODEADDRLOOKUP TRUE</pre>

ENABLEPOSUSERPRIORITY	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , the scheduler will allow users to specify positive job priority values which will be honored. In other words, users can specify a priority that falls in the range of -1024 to +1023, inclusive. If set to <i>FALSE</i> (the default), user priority values are given an upper bound of '0' when users request a positive priority. See <a href="#">USERPRIOWEIGHT</a> .
<b>Example</b>	<pre>ENABLEPOSUSERPRIORITY TRUE</pre> <p><i>Users may now specify positive job priorities and have them take effect (e.g. <code>msub -p 100 job.cmd</code>).</i></p>

ENABLESPVIOLATIONPREEMPTION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , Moab will allow jobs within the same <a href="#">class/queue</a> to <a href="#">preempt</a> when the preemptee is violating a <i>soft</i> <a href="#">usage policy</a> and the preemptor is not.
<b>Example</b>	<pre>ENABLESPVIOLATIONPREEMPTION TRUE</pre>

ENABLEVMDESTROY	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If set to <i>TRUE</i> , enables the automatic destruction of a VM when the VM wall time is expired or when the VM is stale and configured to be destroyed (for more information, see <a href="#">VMSTALEACTION</a> ).
<b>Example</b>	<pre>ENABLEVMDESTROY TRUE</pre>

ENFORCEACCOUNTACCESS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether or not Moab will enforce account access constraints without an allocation manager.
<b>Example</b>	<pre>ENFORCEACCOUNTACCESS TRUE</pre>

ENFORCEGRESACCESS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If a user submits a job with a non-existent gres (e.g. in the case of a typo) and <b>ENFORCEGRESACCESS</b> is not set in <code>moab.cfg</code> , or is set to <i>FALSE</i> , then the requested gres will be created (but will not exist on any nodes) and the job will be deferred (similar to <b>ENFORCEACCOUNTACCESS</b> ).
<b>Example</b>	<pre>ENFORCEGRESACCESS TRUE</pre>

EVENTSERVER	
<b>Format</b>	<HOST>[:<PORT>]
<b>Default</b>	---
<b>Description</b>	Specifies the interface for the event server.
<b>Example</b>	<pre>EVENTSERVER calli3.icluster.org:4702</pre>

FEATURENODETYPEHEADER	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the header used to specify node type via node features (i.e. LL features or PBS node attributes).
<b>Example</b>	<pre>FEATURENODETYPEHEADER xnt</pre> <p><i>Moab will interpret all node features with the leading string <b>xnt</b> as a nodetype specification - as used by the allocation manager and other allocation managers, and assign the associated value to the node. i.e., xntFast.</i></p>

FEATUREPARTITIONHEADER	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the header used to specify node partition via node features (i.e. LL features or PBS node attributes).
<b>Example</b>	<pre>FEATUREPARTITIONHEADER xpt</pre> <p><i>Moab will interpret all node features with the leading string <b>xpt</b> as a partition specification and assign the associated value to the node. i.e., xptGold.</i></p>

FEATUREPROCSPEEDHEADER	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the header used to extract node processor speed via node features (i.e., LL features or PBS node attributes). <b>Note:</b> Adding a trailing '\$' character will specify that only features with a trailing number be interpreted. For example, the header 'sp\$' will match 'sp450' but not 'sport'.
<b>Example</b>	<pre>FEATUREPROCSPEEDHEADER xps</pre> <p><i>Moab will interpret all node features with the leading string <b>xps</b> as a processor speed specification and assign the associated value to the node. i.e., xps950.</i></p>

FEATURERACKHEADER	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the header used to extract node rack index via node features (i.e., LL features or PBS node attributes). <b>Note:</b> Adding a trailing '\$' character will specify that only features with a trailing number be interpreted. For example, the header 'rack\$' will match 'rack4' but not 'racket'.
<b>Example</b>	<pre>FEATURERACKHEADER rack</pre> <p><i>Moab will interpret all node features with the leading string <b>rack</b> as a <b>rack</b> index specification and assign the associated value to the node. i.e., rack16.</i></p>

FEATURESLOTHEADER	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the header used to extract node slot index via node features (i.e., LL features or PBS node attributes). <b>Note:</b> Adding a trailing '\$' character will specify that only features with a trailing number be interpreted. For example, the header 'slot\$' will match 'slot12' but not 'slotted'.
<b>Example</b>	<pre>FEATURESLOTHEADER slot</pre> <p><i>Moab will interpret all node features with the leading string <code>slot</code> as a slot index specification and assign the associated value to the node. i.e., slot16.</i></p>

FEEDBACKPROGRAM	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the name of the program to be run at the completion of each job. If not fully qualified, Moab will attempt to locate this program in the 'tools' subdirectory.
<b>Example</b>	<pre>FEEDBACKPROGRAM /var/moab/fb.pl</pre> <p><i>Moab will run the specified program at the completion of each job.</i></p>

FILEREQUESTISJOBCENTRIC	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether a job's file request is a total request for the job or a per task request.
<b>Example</b>	<pre>FILEREQUESTISJOBCENTRIC TRUE</pre> <p><i>Moab will treat file requests as a total request per job.</i></p>

FILTERCMDFILE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	<p>Running the <a href="#">msub</a> command performs the following operations on the submission script:</p> <ul style="list-style-type: none"> <li>• Replace all comments with spaces (excludes Resource Manager directives)</li> <li>• Strip empty lines</li> <li>• Replace \r with \n</li> <li>• Lock job to a PBS resource manager if \$PBS is found in the script</li> </ul> <p>Include the <b>FILTERCMDFILE</b> parameter in the <code>moab.cfg</code> file that resides on the clients.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF;"> <p> <b>FILTERCMDFILE</b> must be <i>FALSE</i> for <a href="#">REJECTDOSSCRIPTS</a> on page 1026 to work correctly.</p> </div>
<b>Example</b>	<pre>FILTERCMDFILE FALSE</pre> <p><i>Running the msub command does not perform the actions detailed earlier.</i></p>

FORCENODEREPROVISION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	<p>When set to <i>TRUE</i>, this config option causes Moab to reprovision a node, even if it is to the same operating system (in essence rewriting the OS).</p>
<b>Example</b>	<pre>FORCENODEREPROVISION TRUE</pre>

FORCERSVSUBTYPE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies that admin reservations must have a <a href="#">subtype</a> associated with them.
<b>Example</b>	<pre>FORCERSVSUBTYPE TRUE</pre> <p><i>Moab will require all admin reservations to include a subtype.</i></p>

FREETIMELOOKAHEADDURATION	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	<i>2 Months</i>
<b>Description</b>	Specifies how far ahead Moab will look when calculating free time on a node.
<b>Example</b>	<pre>FREETIMELOOKAHEADDURATION 7:00:00:00</pre> <p><i>Moab will look <b>1 week</b> ahead when it calculates free time on a node.</i></p>

FSACCOUNTWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>0</i>
<b>Description</b>	Specifies the weight assigned to the account subcomponent of the fairshare component of priority.
<b>Example</b>	<pre>FSACCOUNTWEIGHT 10</pre>

FSCAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum allowed absolute value for a job's total pre-weighted fairshare component.
<b>Example</b>	<pre>FSCAP 10.0</pre> <p><i>Moab will bind a job's pre-weighted fairshare component by the range +/- 10.0.</i></p>

FSCCLASSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to the class subcomponent of the fairshare component of priority.
<b>Example</b>	<pre>FSCCLASSWEIGHT 10</pre>

FSDECAY	
<b>Format</b>	<DOUBLE>
<b>Default</b>	1.0
<b>Description</b>	Specifies decay rate applied to past fairshare interval when computing effective fairshare usage. Values may be in the range of 0.01 to 1.0. A smaller value causes more rapid decay causing <i>aged</i> usage to contribute less to the overall effective fairshare usage. A value of 1.0 indicates that no decay will occur and all fairshare intervals will be weighted equally when determining effective fairshare usage. See <a href="#">Fairshare Overview</a> .
<b>Example</b>	<pre>FSPOLICY    DEDICATEDPS FSDECAY     0.8 FSDEPTH     8</pre> <p><i>Moab will apply a decay rate of 0.8 to all fairshare windows.</i></p>

FSDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	8
<b>Description</b>	<b>Note:</b> The number of available fairshare windows is bounded by the MAX_FSDEPTH value (32 in Moab). See <a href="#">Fairshare Overview</a> .
<b>Example</b>	<code>FSDEPTH 12</code>

FSENABLECAPRIORITY	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Fairshare priority will increase to target and stop.
<b>Example</b>	<code>FSENABLECAPRIORITY TRUE</code>

FSGROUPWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	
<b>Example</b>	<code>FSGROUPWEIGHT 4</code>

FSINTERVAL	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>12:00:00</code>
<b>Description</b>	Specifies the length of each fairshare <a href="#">window</a> .
<b>Example</b>	<pre>FSINTERVAL 12:00:00</pre> <p><i>Track fairshare usage in 12 hour blocks.</i></p>

FSJPUWEIGHT	
<b>Format</b>	<code>&lt;INTEGER&gt;</code>
<b>Default</b>	<code>0</code>
<b>Description</b>	Specifies the fairshare weight assigned to jobs per user.
<b>Example</b>	<pre>FSJPUWEIGHT 10</pre>

FSMOSTSPECIFCLIMIT	
<b>Format</b>	<code>&lt;BOOLEAN&gt;</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	When checking policy usage limits in a fairshare tree, if the most specific policy limit is passed then do not check the same policy again at higher levels in the tree. For example, if a user has a MaxProc policy limit then do not check the MaxProc policy limit on the account for this same user.
<b>Example</b>	<pre>FSMOSTSPECIFCLIMIT TRUE</pre>

FSPOLICY	
<b>Format</b>	<POLICY>[*] where <POLICY> is one of the following: <i>DEDICATEDPS</i> , <i>DEDICATEDPS%</i> , <i>DEDICATEDPES</i> , or <i>UTILIZEDPS</i> .
<b>Default</b>	---
<b>Description</b>	<p>Specifies the unit of tracking fairshare usage. The following options are:</p> <ul style="list-style-type: none"> <li>• <i>DEDICATEDPS</i> (dedicated processor seconds delivered) tracks dedicated processor seconds.</li> <li>• <i>DEDICATEDPS%</i> (dedicated processor seconds available) to specify that <a href="#">percentage based fairshare</a> should be used. See <a href="#">Fairshare Overview</a> and <a href="#">Fairshare Consumption Metrics</a> for more information.</li> <li>• <i>DEDICATEDPES</i> (dedicated processor-equivalent seconds delivered) tracks dedicated processor-equivalent seconds.</li> <li>• <i>UTILIZEDPS</i> (utilized processor seconds delivered) tracks the number of utilized processor seconds.</li> </ul>
<b>Example</b>	<pre>FSPOLICY DEDICATEDPES</pre> <p><i>Moab will track fairshare usage by dedicated processor-equivalent seconds.</i></p>

FSPPUWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the fairshare weight assigned to processors per user.
<b>Example</b>	<pre>FSPPUWEIGHT 10</pre>

FSPSPUWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the fairshare weight assigned to processor-seconds per user.
<b>Example</b>	<code>FSPSPUWEIGHT 10</code>

FSQOSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight assigned to the QOS fairshare subcomponent.
<b>Example</b>	<code>FSQOSWEIGHT 16</code>

FSTARGETISABSOLUTE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether Moab should base fairshare targets off of delivered cycles or up/available cycles.
<b>Example</b>	<code>FSTARGETISABSOLUTE TRUE</code>

FSTREE	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">SHARES</a> or <a href="#">MEMBERLIST</a>
<b>Default</b>	---
<b>Description</b>	Specifies the share tree distribution for job fairshare prioritization (see <a href="#">Hierarchical Share Tree Overview</a> ).
<b>Example</b>	<pre>FSTREE [geo] SHARES=16 MEMBERLIST=geo103,geo313,geo422</pre>

FSTREEACLPOLICY	
<b>Format</b>	<i>OFF, PARENT, or FULL</i>
<b>Default</b>	<i>FULL</i>
<b>Description</b>	Specifies how Moab should interpret credential membership when building the FSTREE (see <a href="#">Hierarchical Share Tree Overview</a> ).
<b>Example</b>	<pre>FSTREEACLPOLICY PARENT</pre> <i>Credentials will be given access to their parent node when applicable.</i>

FSTREEISREQUIRED	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether a job must have an applicable node in the partition's FSTREE in order to execute within that partition (see <a href="#">Hierarchical Share Tree Overview</a> ).
<b>Example</b>	<pre>FSTREEISREQUIRED TRUE</pre> <i>Jobs must have an applicable node in the FSTREE in order to execute.</i>

FSTREEUSERISREQUIRED	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether the user must be given explicit access to a branch in the FSTREE (see <a href="#">Hierarchical Share Tree Overview</a> ).
<b>Example</b>	<pre>FSTREEUSERISREQUIRED TRUE</pre> <p><i>Users must be given explicit access to FSTREE nodes in order to gain access to the FSTREE.</i></p>

FSUSERWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight assigned to the user fairshare subfactor.
<b>Example</b>	<pre>FSUSERWEIGHT 8</pre>

FSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the priority weight assigned to the summation of the fairshare subfactors (see <a href="#">Priority Factor</a> and <a href="#">Fairshare</a> overviews).
<b>Example</b>	<pre>FSWEIGHT 500</pre>

GEVENTCFG[<GEVENT>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is <b>ACTION</b> , <b>ECOUNT</b> , <b>REARM</b> , or <b>SEVERITY</b> . See <a href="#">Responding to Generic Events</a> for details on values you can assign to each attribute.
<b>Default</b>	---
<b>Description</b>	Specifies how the scheduler should behave when various cluster events are detected. See the <a href="#">Generic Events Overview</a> for more information.
<b>Example</b>	<pre>GEVENTCFG[hitemp] ACTION=avoid,record,notify REARM=00:10:00 GEVENT[nodeerror] SEVERITY=3</pre> <p><i>If a hitemp event is detected, Moab adjusts the node allocation policy to minimize the allocation of the node. Moab also sends emails to cluster administrators and reports the event in the Moab event log.</i></p>

GRES CFG[<GRES>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <b>TYPE</b> , <b>PRIVATE</b> , <b>INVERTTASKCOUNT</b> , <a href="#">FEATUREGRES</a> , or <a href="#">STARTDELAY</a>
<b>Default</b>	---
<b>Description</b>	<p>Specifies associations of generic resources into resource groups.</p> <p>When <b>PRIVATE</b> is set to <b>TRUE</b>, Moab puts the requested generic resource on a separate job request.</p> <p>By default a private request is a request with 1 task with X number of generic resources per task. When <b>INVERTTASKCOUNT</b> and <b>PRIVATE</b> are set to <b>TRUE</b>, Moab makes the generic resource's private request a request with X number of tasks with 1 generic resource per task.</p> <p>See <a href="#">12.6 Managing Consumable Generic Resources</a> for more information.</p>
<b>Example</b>	<pre>GRES CFG[scsi1] TYPE=fastio GRES CFG[scsi2] TYPE=fastio GRES CFG[scsi3] TYPE=fastio</pre> <p><i>The generic resources scsi1, scsi2, and scsi3 are all associated with the generic resource type fastio.</i></p>

GRESTOJOBATTR	
<b>Format</b>	Comma delimited list of generic resources
<b>Default</b>	---
<b>Description</b>	The list of generic resources will also be interpreted as JOB features. See <a href="#">Managing Reservations</a> .
<b>Example</b>	<pre>GRESTOJOBATTR matlab,ccs</pre> <p><i>Jobs which request the generic resources <b>matlab</b> or <b>ccs</b> will have a corresponding job attribute assigned to them.</i></p>

GROUPCFG[<GROUPID>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">General Credential Flags</a> , <b>PRIORITY</b> , <a href="#">ENABLEPROFILING</a> , <b>QLIST</b> , <b>QDEF</b> , <b>PLIST</b> , <b>FLAGS</b> , <a href="#">usage limits</a> , or a <a href="#">fairshare usage limit</a> specification.
<b>Default</b>	---
<b>Description</b>	Specifies group specific attributes. See the <a href="#">flag overview</a> for a description of legal flag values.
<b>Example</b>	<pre>GROUPCFG[staff] MAXJOB=50 QDEF=highprio</pre> <p><i>Up to 50 jobs submitted by members of the group staff will be allowed to execute simultaneously and will be assigned the QOS <b>highprio</b> by default.</i></p>

GROUPWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the priority weight assigned to the specified group priority (See <a href="#">Credential (CRED) Factor</a> ).
<b>Example</b>	<pre>GROUPWEIGHT 20</pre>

GUARANTEEDPREEMPTION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	<p>Causes Moab to lock <a href="#">PREEMPTOR</a> jobs until <a href="#">JOBRETRYTIME</a> expires (essentially, waiting for the <a href="#">PREEMPTEE</a> jobs to finish).</p> <p>It may take some time for the <a href="#">PREEMPTEE</a> jobs to clear out. During that time, the <a href="#">PREEMPTOR</a> job might want to look elsewhere to run, which would be disruptive as it might preempt another set of jobs. If you wish it prevent this, it is recommended that you set <b>GUARANTEEDPREEMPTION</b> to <b>TRUE</b>.</p> <p>For related information, see <a href="#">About preemption</a>, <a href="#">Reservation Policies</a>, <a href="#">DEFERSTARTCOUNT</a>, <a href="#">DEFERTIME</a>, <a href="#">RESERVATIONRETRYTIME</a>, <a href="#">NODEFAILURERESERVETIME</a>, and <a href="#">JOBRETRYTIME</a>.</p>
<b>Example</b>	<pre>GUARANTEEDPREEMPTION TRUE</pre>

HALOCKCHECKTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	9
<b>Description</b>	<p>Specifies how frequently the secondary server checks the timestamp on the lock file. See <a href="#">High Availability Overview</a> for more info.</p>
<b>Example</b>	<pre>HALOCKCHECKTIME 00:00:15</pre> <p><i>The Moab fallback server will check the health of the Moab primary server every 15 seconds.</i></p>

HALOCKUPDATETIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	3
<b>Description</b>	Specifies how frequently the primary server checks the timestamp on the lock file. See <a href="#">High Availability Overview</a> for more info.
<b>Example</b>	<pre>HALOCKUPDATETIME 00:00:03</pre> <p><i>The Moab primary server will check the timestamp of the lock file every 3 seconds.</i></p>

HIDEVIRTUALNODES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	Enables VM management; also used to reveal hypervisors.
<b>Example</b>	<pre>HIDEVIRTUALNODES TRANSPARENT</pre>

IDCFG[X]	
<b>Format</b>	One or more of the following attribute/value pairs: <i>BLOCKEDCREDLIST</i> , <a href="#">CREATECRED</a> , <a href="#">CREATECREDURL</a> , <i>REFRESHPERIOD</i> , <i>RESETCREDLIST</i> or <i>SERVER</i> .
<b>Default</b>	---
<b>Description</b>	<p>This parameter enables the <a href="#">identity manager</a> interface allowing credential, policy, and usage information to be shared with an external information service.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; background-color: #E6F2FF;"> <p><b>i</b> Only one identity manager can be configured at a time.</p> </div>
<b>Example</b>	<pre>IDCFG[info] SERVER=exec://dbquery.pl REFRESHPERIOD=hour</pre> <p><i>Moab will refresh credential info every hour using the specified script.</i></p>

IGNOREMDATASTAGING	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	When set to <i>TRUE</i> , Moab will ignore any resource manager specific data staging on a job and assume the resource manager is processing the request. Currently, this only applies to PBS.
<b>Example</b>	<pre>IGNORERMDATASTAGING TRUE</pre>

IGNORECLASSES	
<b>Format</b>	[!]<CLASS>[,<CLASS>]...
<b>Default</b>	---
<b>Description</b>	By default, if using the TORQUE resource manager, jobs from all listed classes are ignored and not scheduled, tracked, or otherwise processed by Moab. If the <b>not</b> (i.e., '!') character is specified, only jobs from listed classes are processed. See the <a href="#">Moab Side-by-Side Analysis</a> for more information.
<b>Example</b>	<pre>IGNORECLASSES dque, batch</pre> <i>Moab will ignore jobs from classes <i>dque</i> and <i>batch</i>.</i>

IGNOREJOBS	
<b>Format</b>	[!]<JOBID>[,<JOBID>]...
<b>Default</b>	---
<b>Description</b>	By default, listed jobs are ignored and not scheduled, tracked, or otherwise processed by Moab. If the <b>not</b> (i.e., '!') character is specified, only listed jobs are processed. See the <a href="#">Moab Side-by-Side Analysis</a> for more information.
<b>Example</b>	<pre>IGNOREJOBS !14221, 14223</pre> <i>Moab will ignore jobs all classes except <i>14221</i> and <i>14223</i>.</i>

IGNORENODES	
<b>Format</b>	<code>[!]&lt;NODE&gt;[,&lt;NODE&gt;]...</code>
<b>Default</b>	---
<b>Description</b>	By default, all listed nodes are ignored and not scheduled, tracked, or otherwise processed by Moab. If the <b>not</b> (i.e., '!') character is specified, only listed nodes are processed. See the <a href="#">Moab Side-by-Side Analysis</a> for more information.
<b>Example</b>	<pre>IGNORENODES !host3,host4</pre> <p><i>Moab will only process nodes <b>host3</b> and <b>host4</b>.</i></p>

IGNOREPREEMPTPRIORITY	
<b>Format</b>	<code>&lt;BOOLEAN&gt;</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	By default, preemptor jobs can only <a href="#">preempt</a> preemptee jobs if the preemptor has a higher job <a href="#">priority</a> than the preemptee. When this parameter is set to true, the priority constraint is removed allowing any preemptor to preempt any preemptees once it reaches the top of the eligible job queue.
<b>Example</b>	<pre>IGNOREPREEMPTPRIORITY TRUE</pre> <p><i>A preemptor job can preempt any preemptee jobs when it is at the top of the eligible job queue.</i></p>

IGNOREUSERS	
<b>Format</b>	<code>[!]&lt;USERNAME&gt;[,&lt;USERNAME&gt;]...</code>
<b>Default</b>	---
<b>Description</b>	By default, if using the TORQUE resource manager, jobs from all listed users are ignored and not scheduled, tracked, or otherwise processed by Moab. If the <b>not</b> (i.e., '!') character is specified, only jobs from listed users are processed. (See the <a href="#">Moab Side-by-Side Analysis</a> for more information.)
<b>Example</b>	<pre>IGNOREUSERS testuser1,annapolis</pre> <p><i>Moab will ignore jobs from users <code>testuser1</code> and <code>annapolis</code>.</i></p>

#INCLUDE	
<b>Format</b>	<code>&lt;STRING&gt;</code>
<b>Default</b>	---
<b>Description</b>	Specifies another file which contains more configuration parameters. If <code>&lt;STRING&gt;</code> is not an absolute path, Moab will search its home directory for the file.
<b>Example</b>	<pre>#INCLUDE moab.acct</pre> <p><i>Moab will process the parameters in <code>moab.acct</code> as well as <code>moab.cfg</code></i></p>

INSTANTSTAGE	
<b>Format</b>	<code>&lt;BOOLEAN&gt;</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	<b>Deprecated.</b> Use <a href="#">JOBMIGRATEPOLICY</a> . Specifies whether Moab should instantly stage jobs to the underlying resource manager when a job is submitted through <a href="#">msub</a> .
<b>Example</b>	<pre>INSTANTSTAGE TRUE</pre>

INVALIDFSTREEMSG	
<b>Format</b>	"<STRING>"
<b>Default</b>	<i>"no valid fstree node found"</i>
<b>Description</b>	Specifies the error message that should be attached to jobs that cannot run because of a fairshare tree configuration violation.
<b>Example</b>	<pre>INVALIDFSTREEMSG "account is invalid for requested partition"</pre>

JOBACTIIONONNODEFAILURE	
<b>Format</b>	<a href="#">CANCEL</a> on page 409, <a href="#">FAIL</a> on page 409, <a href="#">HOLD</a> on page 409, <a href="#">IGNORE</a> on page 409, <a href="#">NOTIFY</a> on page 409, or <a href="#">REQUEUE</a> on page 409
<b>Default</b>	---
<b>Description</b>	<p>Specifies the action to take if Moab detects that a node allocated to an active job has failed (state is <a href="#">down</a>). By default, Moab only reports this information via diagnostic commands. If this parameter is set, Moab will cancel or requeue the active job. See <a href="#">Reallocating Resources When Failures Occur</a> for more information.</p> <p><b>Note:</b> The <i>HOLD</i> value is only applicable when using <a href="#">checkpointing</a>.</p>
<b>Example</b>	<pre>JOBACTIIONONNODEFAILURE REQUEUE</pre> <p><i>Moab will requeue active jobs which have allocated nodes which have failed during the execution of the job.</i></p>

JOBAGGREGATIONTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0
<b>Description</b>	Specifies the minimum amount of time the scheduler should wait after receiving a job event until it should process that event. This parameter allows sites with <i>bursty</i> job submissions to process job events in groups decreasing total job scheduling cycles and allowing the scheduler to make more intelligent choices by aggregating job submissions and choosing between the jobs. See <a href="#">Considerations for Large Clusters</a> .
<b>Example</b>	<pre>JOBAGGREGATIONTIME 00:00:04 RMPOLLINTERVAL 30,30</pre> <p><i>Moab will wait 4 seconds between scheduling cycles when job events have been received and will wait 30 seconds between scheduling cycles otherwise.</i></p>

JOB CFG	
<b>Format</b>	<ATTR>=<VAL> where <ATTR> is one of <a href="#">FLAGS</a> , <a href="#">GRES</a> , <a href="#">NODERANGE</a> , <a href="#">PRIORITYF</a> , <a href="#">PROC RANGE</a> , <a href="#">QOS</a> , <a href="#">RARCH</a> , <a href="#">RFEATURES</a> , <a href="#">ROPSYS</a> , <a href="#">SELECT</a> , or <a href="#">TARGETLOAD</a>
<b>Default</b>	---
<b>Description</b>	<p>Specifies attributes for jobs which satisfy the specified profile. The <b>SELECT</b> attribute allows users to specify the job template by using <a href="#">msub -l template=</a>.</p> <p>The <b>JOB CFG</b> parameter supports the following attributes:  <b>NONE, ACCOUNT, ACTION, AUTOSIZE, CLASS, CPULIMIT, DESCRIPTION, DGRES, FAILUREPOLICY, GROUP, IFLAGS, JOBSRIPT MEM</b> (for <b>MEM=&lt;value&gt;</b>), <b>MEMORY</b> (for <b>MEMORY=\$LEARN</b>), <b>NODEACCESSPOLICY, NODEMOD, PARTITION, PREF, QOS, RESTARTABLE, RM, RMSERVICEJOB, SELECT, SOFTWARE, SRM, TEMPLIMIT, TFLAGS, USER, VMUSAGE, WALLTIME, WORK</b></p> <p>It also supports the following Wiki attributes:  <b>ARGS, DMEM, DDISK, DWAP, ERROR, EXEC, EXITCODE, GATTR, GEVENT, IWD, JNAME, NAME, PARTITIONMASK, PRIORITYF, RDISK, RSWAP, RAGRES, RCGRES, TASKPERNODE, TRIGGER, VARIABLE, NULL</b></p> <p><b>Note:</b> The <i>index</i> to the <b>JOB CFG</b> parameter can either be an admin-chosen <a href="#">job template</a> name or the exact name of job reported by one or more <a href="#">workload</a> queries. See <a href="#">Wiki Attributes</a> and <a href="#">Job Template Extensions</a>.</p>
<b>Example</b>	<pre>JOB CFG[sql] RFEATURES=sqlnode QOS=service</pre> <p><i>When the sql job is detected, it will have the specified default qos and node feature attributes set.</i></p>

JOB CPURGETIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	00:05:00
<b>Description</b>	Specifies the amount of time Moab will preserve detailed information about a completed job (see <a href="#">showq -c</a> and <a href="#">checkjob</a> ).
<b>Example</b>	<pre>JOB CPURGETIME 02:00:00</pre> <p><i>Moab will maintain detailed job information for 2 hours after a job has completed.</i></p>

JOBTRUNCATENLCP	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Specifies whether Moab will store only the first node of the node list for a completed job in the checkpoint file.
<b>Example</b>	<pre>JOBTRUNCATENLCP TRUE</pre> <p><i>JOBTRUNCATENLCP reduces the amount of memory Moab uses to store completed job information.</i></p>

JOBEXTENDSTARTWALLTIME	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	Extends the job walltime when Moab starts the job up to the lesser of the maximum or the next reservation (rounded down to the nearest minute).
<b>Example</b>	<pre>JOBEXTENDSTARTWALLTIME TRUE</pre> <p>Submit job with a minimum wallclock limit and a walltime; for example:</p> <pre>echo sleep 500   msub -A ee -l nodes=5,minwclimit=5:00,walltime=30:00,partition=g02</pre> <p><i>At job start, Moab recognizes the nodes assigned to the specified job and extends the walltime for the job (one time at job start) up to the lesser of the maximum walltime requested or the least amount of time available for any of the nodes until the next reservation on that node.</i></p>

JOBFAILRETRYCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	<p>Specifies the number of times a job is requeued and restarted by Moab if the job fails (if the job itself returns a non-zero exit code). Some types of jobs may succeed if automatically retried several times in short succession. This parameter was created with these types of jobs in mind. Note that the job in question must also be restartable (the job needs to have the "RESTARTABLE" flag set on it) and the RM managing the job must support requeuing and starting completed jobs. If a job fails too many times, and reaches the number of retries given by JobFailRetryCount, then a UserHold is placed on the job and a message is attached to it signifying that the job has a "restart count violation."</p>
<b>Example</b>	<pre>JOBFAILRETRYCOUNT 7</pre> <p><i>Any job with a RESTARTABLE flag is requeued, if it fails, up to 7 times before a UserHold is placed on it.</i></p>

JOBIDWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	<p>Specifies the weight to be applied to the job's id. See <a href="#">Attribute (ATTR) Factor</a>.</p>
<b>Example</b>	<pre>JOBIDWEIGHT -1</pre> <p><i>Later jobs' priority will be negatively affected.</i></p>

JOBMATCHCFG	
<b>Format</b>	<ATTR>=<VAL> where <ATTR> is one of <a href="#">JMIN</a> , <a href="#">JMAX</a> , <a href="#">JDEF</a> , <a href="#">JSET</a> , or <a href="#">JSTAT</a>
<b>Default</b>	---
<b>Description</b>	Specifies the job templates which must be matched and which will be applied in the case of a match.
<b>Example</b>	<pre>JOBMATCHCFG[sql] JMIN=interactive JSTAT=istat</pre>

JOBMAXHOLDTIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	---
<b>Description</b>	Specifies the amount of time a job can be held before it is canceled automatically.
<b>Example</b>	<pre>JOBMAXHOLDTIME 02:00:00</pre> <p><i>Moab will keep jobs in any HOLD state for 2 hours before canceling them.</i></p>

JOBMAXNODECOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1024
<b>Description</b>	Specifies the maximum number of nodes which can be allocated to a job. After changing this parameter, Moab must be restarted. <b>Note:</b> This value cannot exceed either <b>MMAX_NODE</b> or <b>MMAX_TASK_PER_JOB</b> . If larger values are required, these values must also be increased. Moab must be restarted before changes to this command will take effect. The command <code>mdiag -S</code> will indicate if any job node count overflows have occurred. See <a href="#">Consideration for Large Clusters</a> .
<b>Example</b>	<pre>JOBMAXNODECOUNT 4000</pre>

JOBMAXOVERRUN	
<b>Format</b>	[[[DD:]HH:]MM:]SS],[[DD:]HH:]MM:]SS
<b>Default</b>	(no soft limit), <i>10 minutes</i> (hard limit)
<b>Description</b>	<p>Soft and hard limit of the amount of time Moab will allow a job to exceed its wallclock limit before it first sends a mail to the primary admin (soft limit) and then terminates the job (hard limit). See <a href="#">WCVIOLATIONACTION</a> or <a href="#">Usage-based Limits</a>.</p> <div style="border: 1px solid #0070C0; padding: 10px; margin-top: 10px;"> <p><b>i</b> If you run Moab with the TORQUE resource manager, you must set the <b>signwalltime</b> parameter to <i>true</i> in the <code>/var/spool/torque/mom_priv/config</code> file, otherwise the <code>pbs_mom</code> will kill any job that exceeds its walltime. See the <a href="#">TORQUE documentation</a> for more information.</p> </div>
<b>Example</b>	<pre>JOBMAXOVERRUN 15:00,1:00:00</pre> <p><i>Jobs may exceed their wallclock limit by up to 1 hour, but Moab will send an email to the primary administrator when a job exceeds its walltime by 15 minutes.</i></p>

JOBMAXPREEMPTPERITERATION	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (No Limit)
<b>Description</b>	Maximum number of jobs allowed to be preempted per iteration.
<b>Example</b>	<pre>JOBMAXPREEMPTPERITERATION 10</pre>

JOBMAXSTARTPERITERATION	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (No Limit)
<b>Description</b>	Maximum number of jobs allowed to start per <a href="#">iteration</a> .
<b>Example</b>	<pre>JOBMAXSTARTPERITERATION 10</pre>

JOBMAXSTARTTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	-1 (NO LIMIT)
<b>Description</b>	length of time a job is allowed to remain in a 'starting' state. If a 'started' job does not transition to a running state within this amount of time, Moab will cancel the job, believing a system failure has occurred.
<b>Example</b>	<pre>JOBMAXSTARTTIME 2:00:00</pre> <p><i>Jobs may attempt to start for up to 2 hours before being canceled by the scheduler</i></p>

JOBMAXTASKCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	4096
<b>Description</b>	Specifies the total number of tasks allowed per job.
<b>Example</b>	<pre>JOBMAXTASKCOUNT 226000</pre>

JOBMIGRATEPOLICY	
<b>Format</b>	One of the following: <i>IMMEDIATE</i> , <i>JUSTINTIME</i> , or <i>AUTO</i>
<b>Default</b>	<i>AUTO</i>
<b>Description</b>	Upon using the <a href="#">msub</a> command to submit a job, you can allocate the job to immediately ( <i>IMMEDIATE</i> ) migrate to the resource manager, or you can instruct Moab to only migrate the job to the resource manager when it is ready to run ( <i>JUSTINTIME</i> ). Specifying <i>AUTO</i> allows MOAB to determine on a per-job basis whether to use <i>IMMEDIATE</i> or <i>JUSTINTIME</i> .
<b>Example</b>	<pre>JOBMIGRATEPOLICY JUSTINTIME</pre>

JOBNAMEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Specifies the weight to be applied to the job's name if the Name contains an integer. See <a href="#">Attribute (ATTR) Factor</a> .
<b>Example</b>	<pre>JOBNAMEWEIGHT 1</pre>

JOBNODEMATCHPOLICY	
<b>Format</b>	<i>EXACTNODE</i> or <i>EXACTPROC</i>
<b>Default</b>	---
<b>Description</b>	Specifies additional constraints on how compute nodes are to be selected. <i>EXACTNODE</i> indicates that Moab should select as many nodes as requested even if it could pack multiple tasks onto the same node. <i>EXACTPROC</i> indicates that Moab should select only nodes with exactly the number of processors configured as are requested per node even if nodes with excess processors are available.
<b>Example</b>	<pre>JOBNODEMATCHPOLICY EXACTNODE</pre> <p><i>In a PBS/Native job with resource specification nodes=&lt;x&gt;:ppn=&lt;y&gt;, Moab will allocate exactly &lt;y&gt; task on each of &lt;x&gt; distinct nodes.</i></p>

JOBPREEMPTMAXACTIVETIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0
<b>Description</b>	The amount of time in which a job may be eligible for preemption. See <a href="#">Job Preemption</a> .
<b>Example</b>	<pre>JOBPREEMPTMAXACTIVETIME 00:05:00</pre> <p><i>A job is preemptable for the first 5 minutes of its run time.</i></p>

JOBPREEMPTMINACTIVETIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0
<b>Description</b>	The minimum amount of time a job must be active before being considered eligible for preemption. See <a href="#">Job Preemption</a> .
<b>Example</b>	<div style="border: 1px dashed gray; padding: 5px; margin-bottom: 5px;"> JOBPREEMPTMINACTIVETIME 00:05:00 </div> <p><i>A job must execute for <b>5 minutes</b> before Moab will consider it eligible for preemption.</i></p>

JOBPRIOACCRUALPOLICY	
<b>Format</b>	<i>ACCRUE</i> or <i>RESET</i>
<b>Default</b>	<i>ACCRUE</i>
<b>Description</b>	<p>Specifies how Moab should track the dynamic aspects of a job's priority. <i>ACCRUE</i> indicates that the job will accrue queuetime based priority from the time it is submitted unless it violates any of the policies not specified in <a href="#">JOBPRIOEXCEPTIONS</a>. <i>RESET</i> indicates that it will accrue priority from the time it is submitted unless it violates any of the <a href="#">JOBPRIOEXCEPTIONS</a>. However, with <i>RESET</i>, if the job does violate <a href="#">JOBPRIOEXCEPTIONS</a> then its queuetime based priority will be reset to 0.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p><b>i</b> <a href="#">JOBPRIOACCRUALPOLICY</a> is a global parameter, but can be configured to work only in <a href="#">QOSCFG</a>:</p> <pre style="border: 1px dashed #0070C0; padding: 5px; display: inline-block;">QOSCFG[arrays] JOBPRIOACCRUALPOLICY=ACCRUE</pre> </div> <p>The following old <a href="#">JOBPRIOACCRUALPOLICY</a> values have been deprecated and should be adjusted to the following values:</p> <ul style="list-style-type: none"> <li>• <a href="#">QUEUEPOLICY</a>= <i>ACCRUE</i> and <a href="#">JOBPRIOEXCEPTIONS</a><i>SOFTPOLICY,HARDPOLICY</i></li> <li>• <a href="#">QUEUEPOLICYRESET</a>= <i>RESET</i> and <a href="#">JOBPRIOEXCEPTIONS</a><i>SOFTPOLICY,HARDPOLICY</i></li> <li>• <a href="#">ALWAYS</a>= <i>ACCRUE</i> and <a href="#">JOBPRIOEXCEPTIONS</a><i>ALL</i></li> <li>• <a href="#">FULLPOLICY</a>= <i>ACCRUE</i> and <a href="#">JOBPRIOEXCEPTIONS</a><i>NONE</i></li> <li>• <a href="#">FULLPOLICYRESET</a>= <i>RESET</i> and <a href="#">JOBPRIOEXCEPTIONS</a><i>NONE</i></li> </ul>
<b>Example</b>	<pre style="border: 1px dashed #0070C0; padding: 5px; display: inline-block;">JOBPRIOACCRUALPOLICY RESET</pre> <p style="border: 1px dashed #0070C0; padding: 5px; margin-top: 10px;"><i>Moab will adjust the job's dynamic priority subcomponents, i.e., <a href="#">QUEUE</a>TIME, <a href="#">XFACTOR</a>, and <a href="#">TARGET</a>QUEUE TIME, etc. each iteration that the job does not violate any <a href="#">JOBPRIOEXCEPTIONS</a>, if it is found in violation, its queuetime will be reset to 0.</i></p>

JOBPRIOEXCEPTIONS	
<b>Format</b>	Comma delimited list of any of the following: <i>DEFER</i> , <i>DEPENDS</i> , <i>SOFTPOLICY</i> , <i>HARDPOLICY</i> , <i>IDLEPOLICY</i> , <i>USERHOLD</i> , <i>BATCHHOLD</i> , and <i>SYSTEMHOLD</i> ( <i>ALL</i> or <i>NONE</i> can also be specified on their own)
<b>Default</b>	<i>NONE</i>
<b>Description</b>	<p>Specifies exceptions for calculating a job's dynamic priority (QUEUE TIME, XFACTOR, TARGETQUEUE TIME). Normally, when a job violates a policy, is placed on hold, or has an unsatisfied dependency, it will not accrue priority. Exceptions can be configured to allow a job to accrue priority in spite of any of these violations. With <i>DEPENDS</i> a job will increase in priority even if there exists an unsatisfied dependency. With <i>SOFTPOLICY</i>, <i>HARDPOLICY</i>, or <i>IDLEPOLICY</i> a job can accrue priority despite violating a specific limit. With <i>DEFER</i>, <i>USERHOLD</i>, <i>BATCHHOLD</i>, or <i>SYSTEMHOLD</i> a job can accrue priority despite being on hold.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p> <b>JOBPRIOEXCEPTIONS</b> is a global parameter, but can be configured to work only in <a href="#">QOSCFG</a>:</p> <pre style="border: 1px dashed #ccc; padding: 5px; display: inline-block; margin: 5px 0;">QOSCFG[arrays] JOBPRIOEXCEPTIONS=IDLEPOLICY</pre> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; display: inline-block; margin-bottom: 5px;">JOBPRIOEXCEPTIONS BATCHHOLD, SYSTEMHOLD, DEPENDS</pre> <p style="border: 1px dashed #ccc; padding: 5px; display: inline-block; margin-top: 5px;"><i>Jobs will accrue priority in spite of batchholds, systemholds, or unsatisfied dependencies.</i></p>

JOBPRIOF	
<b>Format</b>	<ATTRIBUTE>[<VALUE>]=<PRIORITY> where <ATTRIBUTE> is one of <b>ATTR</b> , <b>GRES</b> or <b>STATE</b>
<b>Default</b>	---
<b>Description</b>	Specifies attribute priority weights for jobs with specific attributes, generic resource requests, or states. State values must be one of the standard Moab <a href="#">job states</a> . See <a href="#">Attribute-Based Job Prioritization</a> .
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; display: inline-block; margin-bottom: 5px;">JOBPRIOF          STATE[Running]=100  STATE[Suspended]=1000  ATTR[PREEMPTEE]=200 GRES[biocalc]=5 ATTRATTRWEIGHT   1 ATTRSTATEWEIGHT  1</pre> <p style="border: 1px dashed #ccc; padding: 5px; display: inline-block; margin-top: 5px;"><i>Moab will adjust the job's dynamic priority subcomponents.</i></p>

JOBPURGETIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>0</code> (purge immediately if the resource manager does not report the job)
<b>Description</b>	The amount of time Moab will keep a job record which is no longer reported by the resource manager. Useful when using a resource manager which <i>drops</i> information about a job due to internal failures. See <a href="#">JOBPCPURGETIME</a> .
<b>Example</b>	<pre>JOBPURGETIME 00:05:00</pre> <p><i>Moab will maintain a job record for 5 minutes after the last update regarding that object received from the resource manager.</i></p>

JOBREJECTPOLICY	
<b>Format</b>	One or more of <i>CANCEL</i> , <i>HOLD</i> , <i>IGNORE</i> (beta), <i>MAIL</i> , or <i>RETRY</i>
<b>Default</b>	<i>HOLD</i>
<b>Description</b>	Specifies the action to take when the scheduler determines that a job can never run. <i>CANCEL</i> issues a call to the resource manager to cancel the job. <i>HOLD</i> places a <i>batch</i> hold on the job preventing the job from being further evaluated until released by an administrator. ( <b>Note:</b> Administrators can dynamically alter job attributes and possibly <i>fix</i> the job with <a href="#">mjobctl -m</a> .) With <i>IGNORE</i> (currently in beta), the scheduler will allow the job to exist within the resource manager queue but will neither process it nor report it. <a href="#">MAIL</a> will send email to both the admin and the user when rejected jobs are detected. If <i>RETRY</i> is set, then Moab will allow the job to remain idle and will only attempt to start the job when the policy violation is resolved. Any combination of attributes may be specified. See <a href="#">QOSREJECTPOLICY</a> .
<b>Example</b>	<pre>JOBREJECTPOLICY MAIL,CANCEL</pre>

JOBREMOVEENVVARLIST	
<b>Format</b>	Comma-delimited list of strings
<b>Default</b>	---
<b>Description</b>	<p>Moab will remove the specified environment variables from the job's environment before migrating the job to its destination resource manager. This is useful when jobs submit themselves from one cluster to another with the full environment.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;">  This parameter is currently only supported with TORQUE resource managers.                 </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 5px;"> <pre>JOBREMOVEENVVARLIST PBS_SERVER, TZ</pre> </div> <div style="border: 1px dashed #ccc; padding: 10px;"> <p><i>Moab will remove the environment variables <b>PBS_SERVER</b> and <b>TZ</b> before submitting jobs.</i></p> </div>

JOBRETRYTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	00:00:60
<b>Description</b>	<p>Period of time Moab will continue to attempt to start a job which has failed to start due to transient failures or which has successfully started and was then rejected by the resource manager due to transient failures. (For related information, see <a href="#">Reservation Policies</a>, <a href="#">DEFERSTARTCOUNT</a>, <a href="#">DEFERTIME</a>, <a href="#">RESERVATIONRETRYTIME</a>, <a href="#">NODEFAILURERESERVETIME</a>, and <a href="#">GUARANTEEDPREEMPTION</a>.)</p>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 5px;"> <pre>JOBRETRYTIME 00:05:00</pre> </div> <div style="border: 1px dashed #ccc; padding: 10px;"> <p><i>Moab will try for up to <b>5 minutes</b> to restart jobs if the job start has failed due to transient errors.</i></p> </div>

LIMITEDJOBCEP	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether there should be limited job checkpointing (see <a href="#">Consideration for Large Clusters</a> ). With <b>LIMITEDJOBCEP</b> enabled, Moab will only checkpoint a job if it is modified with <a href="#">mjobctl on page 212</a> or if it has been submitted with <a href="#">msub on page 286</a> but has not migrated. In all other cases, Moab does not checkpoint the job and all Moab-specific information (such as messages attached to the job) is lost. No TORQUE-specific information will be lost.
<b>Example</b>	<pre>LIMITEDJOBCEP TRUE</pre> <p><i>Moab will only maintain scheduler checkpoint information for jobs with explicitly modified job attributes. Some minor job performance and usage statistics may be lost.</i></p>

LIMITEDNODECEP	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether there should be limited node checkpointing (see <a href="#">Consideration for Large Clusters</a> ).
<b>Example</b>	<pre>LIMITEDNODECEP TRUE</pre> <p><i>Moab will only maintain scheduler checkpoint information for nodes with explicitly modified job attributes. (some minor node performance and usage statistics may be lost)</i></p>

LOADALLJOB	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether Moab should load, during startup, all non-completed jobs in the checkpoint files regardless of whether or not their corresponding resource managers are active. For example, this allows source peers to continue showing remote jobs in the queue based on checkpointed info, even though the destination peer is offline.
<b>Example</b>	<pre>LOADALLJOB TRUE</pre> <p><i>Moab will load, at startup, all non-completed jobs from all checkpoint files.</i></p>

LOCKFILE	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the path for the lock (pid) file used by Moab.
<b>Example</b>	<pre>LOCKFILE /var/spool/moab/lock</pre>

LOGDIR	
<b>Format</b>	<STRING>
<b>Default</b>	log
<b>Description</b>	Specifies the directory in which log files will be maintained. If specified as a relative path, <b>LOGDIR</b> will be relative to \$(MOABHOMEDIR) See <a href="#">Logging Overview</a> for more information.
<b>Example</b>	<pre>LOGDIR /var/spool/moab</pre> <p><i>Moab will record its log files directly into the /var/spool/moab directory</i></p>

LOGFACILITY	
<b>Format</b>	Colon delimited list of one or more of the following: <i>CORE, SCHED, SOCK, UI, LL, CONFIG, STAT, SIM, STRUCT, FS, CKPT, BANK, RM, PBS, WIKI, ALL</i>
<b>Default</b>	<i>ALL</i>
<b>Description</b>	Specifies which types of events to log (see <a href="#">Logging Overview</a> ).
<b>Example</b>	<pre>LOGFACILITY RM:PBS</pre> <p><i>Moab will log only events involving general resource manager or PBS interface activities.</i></p>

LOGFILE	
<b>Format</b>	<STRING>
<b>Default</b>	<i>moab.log</i>
<b>Description</b>	Name of the Moab log file. This file is maintained in the directory pointed to by <LOGDIR> unless <LOGFILE> is an absolute path (see <a href="#">Logging Overview</a> )
<b>Example</b>	<pre>LOGFILE moab.test.log</pre> <p><i>Log information will be written to the file <i>moab.test.log</i> located in the directory pointed to by the <a href="#">LOGDIR</a> parameter.</i></p>

LOGFILEMAXSIZE	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>10000000</i>
<b>Description</b>	Maximum allowed size (in bytes) of the log file before it will be rolled (see <a href="#">Logging Overview</a> ).
<b>Example</b>	<pre>LOGFILEMAXSIZE 50000000</pre> <p><i>Log files will be rolled when they reach <b>50 MB</b> in size</i></p>

LOGFILEROLLDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	3
<b>Description</b>	Number of old log files to maintain (i.e., when full, moab.log will be renamed moab.log.1, moab.log.1 will be renamed moab.log.2, ...). See <a href="#">Logging Overview</a> .
<b>Example</b>	<pre>LOGFILEROLLDEPTH 5</pre> <p><i>Moab will maintain and roll the last 5 log files.</i></p>

LOGLEVEL	
<b>Format</b>	<INTEGER> (0-9)
<b>Default</b>	0
<b>Description</b>	Specifies the verbosity of Moab logging where 9 is the most verbose ( <b>Note:</b> each logging level is approximately an order of magnitude more verbose than the previous level). See <a href="#">Logging Overview</a> .
<b>Example</b>	<pre>LOGLEVEL 4</pre> <p><i>Moab will write all Moab log messages with a threshold of 4 or lower to the moab.log file.</i></p>

LOGLEVELOVERRIDE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>When this parameter is on, if someone runs a command with <code>--loglevel=&lt;x&gt;</code>, that loglevel, if higher than the current loglevel, is used on the scheduler side for the duration of the command. All logs produced during that time are put into a separate log file (this creates a "gap" in the normal logs). This can be very useful for debugging, but it is recommend that this be used only when diagnosing a specific problem so that users can't affect performance by submitting multiple <code>--loglevel</code> commands.</p> <div style="border: 1px solid #0070c0; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> This parameter does not work with threaded commands (such as <code>showq</code>, <code>mdiag -n</code>, and <code>mdiag -j</code>).</p> </div>
<b>Example</b>	<code>LOGLEVELOVERRIDE TRUE</code>

LOGPERMISSIONS	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>644</i>
<b>Description</b>	Specifies the octal number that represents read, write, and execute permissions.
<b>Example</b>	<div style="border: 1px dashed #ccc; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p><code>LOGPERMISSIONS 600</code></p> <p><i>Allows the file owner to read and write permissions, but denies rights to the group and others.</i></p> </div>

LOGROLLACTION	
<b>Format</b>	< <i>STRING</i> >
<b>Default</b>	---
<b>Description</b>	Specifies a script to run when the logs roll. The script is run as a trigger and can be viewed using <a href="#">mdiag -T</a> . For example, a script can be specified that always moves the first rolled log file, <code>moab.log.1</code> , to an archive directory for longer term storage.
<b>Example</b>	<pre>LOGROLLACTION /usr/local/tools/logroll.pl</pre>

MAILPROGRAM	
<b>Format</b>	[<Full_Path_To_Mail_Command>   <i>DEFAULT</i>   <i>NONE</i> ][@<DEFAULTMAILDOMAIN>]
<b>Default</b>	<i>NONE</i>
<b>Description</b>	<p>If set to <i>NONE</i>, no mail is sent. If set to <i>DEFAULT</i>, Moab sends mail via the system's default mail program (usually <code>/usr/bin/sendmail</code>). If set to the local path of a mail program, Moab uses the specified mail program to send mail.</p> <p>By default, Moab mail notification is disabled. To enable, you must set <b>MAILPROGRAM</b> to <i>DEFAULT</i> or specify some other locally available mail program. If the <i>default mail domain</i> is set, emails will be routed to this domain unless a per-user domain is specified using the <b>EMAILADDRESS</b> attribute of the <a href="#">USERCFG</a> parameter. If neither of these values is set, Moab uses "<code>@localhost</code>" as the mail domain. See <a href="#">Notify Admins</a>.</p> <p>For jobs, the email address used on the <a href="#">msub -M</a> option overrides all other user email addresses. Additionally, administrators are notified in the case of job violations.</p>
<b>Example</b>	<pre>MAILPROGRAM DEFAULT</pre> <p><i>Moab sends mail via the system's default mail program, /usr/bin/sendmail.</i></p> <pre>MAILPROGRAM /usr/local/bin/sendmail@mydomain.com</pre> <p><i>Moab sends mail via the mail program located at /usr/local/bin/sendmail with default mail domain @mydomain.com</i></p>

MAXGRES	
<b>Format</b>	<INTEGER>
<b>Default</b>	512
<b>Description</b>	Specifies how many generic resources Moab should manage.
<b>Example</b>	<code>MAXGRES 1024</code>

MAXGMETRIC	
<b>Format</b>	<INTEGER>
<b>Default</b>	10
<b>Description</b>	Specifies how many generic metrics Moab should manage.
<b>Example</b>	<code>MAXGMETRIC 20</code>

MAXJOB	
<b>Format</b>	<INTEGER>
<b>Default</b>	4096
<b>Description</b>	<p>Specifies the maximum quantity of jobs for which Moab should allocate memory used for tracking jobs. If Moab is tracking the maximum quantity of jobs specified by this parameter, it rejects subsequent jobs submitted by any user since it has no memory left with which to track newly submitted jobs.</p> <p>If a user submitted a job with the <code>msub</code> command, this rejection behavior requires the user to resubmit the job at a later time after other jobs have completed, which frees memory in which Moab can place later-submitted jobs. If a user submitted a job with the TORQUE <code>qsub</code> command, TORQUE will automatically resubmit the job to Moab until Moab accepts it.</p> <p>The <code>mdiag -S</code> command indicates if any job overflows have occurred.</p> <p>If this parameter's value is changed, it does not go into effect until Moab restarts. Moab reads the parameter only on initial startup and uses its value to allocate the memory it uses to track jobs.</p>
<b>Example</b>	<code>MAXJOB 45000</code>

MAXNODE	
<b>Format</b>	<INTEGER>
<b>Default</b>	5120
<b>Description</b>	Specifies the maximum number of compute nodes supported.
<b>Example</b>	<pre>MAXNODE 10000</pre>

MAXRSVPERNODE	
<b>Format</b>	<INTEGER>
<b>Default</b>	48
<b>Description</b>	<p>Specifies the maximum number of reservations on a node.</p> <p>For large SMP systems (&gt;512 processors/node), Adaptive Computing advises adjusting the value to approximately twice the average sum of admin, standing, and job reservations present.</p> <p>A second number, led by a comma, can also be specified to set a maximum number of reservations for nodes that are part of the SHARED partition.</p> <p>The maximum possible value of <b>MAXRSVPERNODE</b> is 8192 for a global node and 4096 for any other node.</p> <p>Moab must be restarted for any changes to this parameter to take effect. The command <code>mdiag -S</code> indicates whether any node reservation overflows have occurred. See <a href="#">Considerations for Large Clusters</a>.</p> <div style="border: 1px solid red; padding: 5px; margin-top: 10px;">  Do not lower the MAXRSVPERNODE value while there are active jobs in the queue. This can lead to queue instability and certain jobs could become stuck or disconnected from the system.         </div>
<b>Example</b>	<pre>MAXRSVPERNODE 64</pre> <p><i>64 is the maximum number of reservations on a single node.</i></p> <pre>MAXRSVPERNODE 100,7000</pre> <p><i>100 is the maximum number of reservations on a single node, and 7000 is the maximum number of reservations for global nodes.</i></p>

MEMREFRESHINTERVAL	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS   job:&lt;COUNT&gt;</code>
<b>Default</b>	---
<b>Description</b>	Specifies the time interval or total job query count at which Moab will perform garbage collection to free memory associated with resource manager API's which possess memory leaks (i.e., Loadleveler, etc.).
<b>Example</b>	<pre># free memory associated with leaky RM API MEMREFRESHINTERVAL 24:00:00</pre> <p><i>Moab will perform garbage collection once every 24 hours.</i></p>

MEMWEIGHT	
<b>Format</b>	<code>&lt;INTEGER&gt;</code>
<b>Default</b>	0
<b>Description</b>	Specifies the coefficient to be multiplied by a job's MEM (dedicated memory in MB) factor. See <a href="#">Resource Priority Overview</a> .
<b>Example</b>	<pre>RESWEIGHT 10 MEMWEIGHT 1000</pre> <p><i>Each job's priority will be increased by <math>10 * 1000 * \text{&lt;request memory&gt;}</math>.</i></p>

MESSAGEQUEUEADDRESS	
<b>Format</b>	The IP address of the machine on which Moab is generating events.
<b>Default</b>	* (all)
<b>Description</b>	When a user subscribes to the events Moab provides and delivers via zeroMQ, s/he must do so by specifying <code>tcp://&lt;ipAddress&gt;:&lt;port&gt;</code> . <b>MESSAGEQUEUEADDRESS</b> specifies the <code>&lt;ipAddress&gt;</code> , which must match the IP address of the machine on which Moab is installed. To specify the port, see <a href="#">MESSAGEQUEUEPORT on page 996</a> .
<b>Example</b>	<pre>MESSAGEQUEUEADDRESS 10.1.0.10</pre> <p><i>To subscribe to Moab events, users must use <code>tcp://10.1.0.10:&lt;port&gt;</code>.</i></p>

MESSAGEQUEUEPORT	
<b>Format</b>	The port of the machine on which Moab is generating events.
<b>Default</b>	5563
<b>Description</b>	When a user subscribes to the events Moab provides and delivers via zeroMQ, s/he must do so by specifying <code>tcp://&lt;ipAddress&gt;:&lt;port&gt;</code> . <b>MESSAGEQUEUEPORT</b> specifies the <code>&lt;port&gt;</code> , which must match the port of the machine on which Moab is installed. To specify the IP address, see <a href="#">MESSAGEQUEUEADDRESS on page 996</a> .
<b>Example</b>	<pre>MESSAGEQUEUEPORT 1010</pre> <p><i>To subscribe to Moab events, users must use <code>tcp://&lt;ipAddress&gt;:1010</code>.</i></p>

MESSAGEQUEUESECRETKEY	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	<p>Causes Moab to encrypt the events delivered via zeroMQ using the Advanced Encryption Standard (AES) algorithm. Must be a <a href="#">Base64</a>-encoded, 128-bit (16-byte) key. Messages will be encrypted using AES in CBC mode where inputs are padded with PKCS5 padding. The initialization vector is calculated by using an MD5 hash of the key specified in <b>MESSAGEQUEUESECRETKEY</b>.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;">  <b>MESSAGEQUEUESECRETKEY</b> can only be specified in the moab-private.cfg file.                 </div>
<b>Example</b>	<pre>MESSAGEQUEUESECRETKEY 1r6RvfqJa6voezy5wAx0hw==</pre>

MINADMINSTIME	
<b>Format</b>	<INTEGER>
<b>Default</b>	60 seconds
<b>Description</b>	Specifies the minimum time a job will be <a href="#">suspended</a> if suspended by an administrator or by a scheduler policy.
<b>Example</b>	<pre>MINADMINSTIME 00:10:00</pre> <div style="border: 1px dashed #000; padding: 10px; margin-top: 10px;"> <p><i>Each job suspended by administrators or policies will stay in the suspended state for at least 10 minutes.</i></p> </div>

MISSINGDEPENDENCYACTION	
<b>Format</b>	<i>CANCEL, HOLD, or RUN</i>
<b>Default</b>	<i>HOLD</i>
<b>Description</b>	Controls what Moab does with a dependent job when its dependency job cannot be found when Moab evaluates the dependent job for scheduling. This only affects jobs whose dependent job cannot be found.
<b>Example</b>	<pre>MISSINGDEPENDENCYACTION CANCEL</pre> <p><i>Any job that has a dependent job that cannot be found is canceled.</i></p>

MSUBQUERYINTERVAL	
<b>Format</b>	<INTEGER>
<b>Default</b>	5 seconds
<b>Description</b>	<p>Specifies the length of the interval (in seconds) between job queries when using <a href="#">msub -K</a>. Jobs submitted with the -K option query the scheduler every <b>MSUBQUERYINTERVAL</b> seconds until the job is completed.</p> <p><b>MSUBQUERYINTERVAL</b> can exist as an environment variable. Any value in <code>moab.cfg</code> overrides the environment variable.</p> <p><b>Note:</b> If <b>MSUBQUERYINTERVAL</b> is set to 0, the -K option will be disabled. Jobs will still submit correctly, but the client will not continue to check on the job.</p>
<b>Example</b>	<pre>MSUBQUERYINTERVAL 60</pre> <p><i>If a user uses the <a href="#">msub -K</a> command, the client remains open and queries the server every 60 seconds until the job completes.</i></p>

NODEACCESSPOLICY	
<b>Format</b>	One of the following: <a href="#">SHARED</a> , <a href="#">SHAREDONLY</a> , <a href="#">SINGLEJOB</a> , <a href="#">SINGLETASK</a> , <a href="#">SINGLEUSER</a> , or <a href="#">UNIQUEUSER</a>
<b>Default</b>	<b>SHARED</b>
<b>Description</b>	Specifies how node resources will be shared by various tasks (See the <a href="#">Node Access Overview</a> for more information).
<b>Example</b>	<pre>NODEACCESSPOLICY SINGLEUSER</pre> <p>Moab will allow resources on a node to be used by more than one job provided that the jobs are all owned by the same user.</p>

NODEALLOCATIONPOLICY	
<b>Format</b>	One of the following: <a href="#">FIRSTAVAILABLE</a> , <a href="#">LASTAVAILABLE</a> , <a href="#">MINRESOURCE</a> , <a href="#">CPULOAD</a> , <a href="#">LOCAL</a> , <a href="#">CONTIGUOUS</a> , <a href="#">MAXBALANCE</a> , <a href="#">PRIORITY</a> , or <a href="#">PLUGIN</a> .
<b>Default</b>	<i>LASTAVAILABLE</i>
<b>Description</b>	<p>Specifies how Moab should allocate available resources to jobs. See <a href="#">Node Allocation Overview</a> for more information.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; margin-top: 10px;"> <p> If <b>ENABLEHIGHTHROUGHPUT</b> on page 950 is <i>TRUE</i>, you must set <b>NODEALLOCATIONPOLICY</b> to <i>FIRSTAVAILABLE</i>.</p> </div>
<b>Example</b>	<pre>NODEALLOCATIONPOLICY MINRESOURCE</pre> <p><i>Moab will apply the node allocation policy <b>MINRESOURCE</b> to all jobs by default.</i></p>

NODEALLOCRESFAILUREPOLICY	
<b>Format</b>	One of the following: <i>CANCEL, HOLD, IGNORE, MIGRATE, NOTIFY, or REQUEUE</i>
<b>Default</b>	<i>NONE</i>
<b>Description</b>	Specifies how Moab should handle active jobs which experience node failures during execution. See the <a href="#">RESFAILPOLICY</a> resource manager extension or the <a href="#">Node Availability Overview</a> .
<b>Example</b>	<pre>NODEALLOCRESFAILUREPOLICY REQUEUE</pre> <p><i>Moab will requeue jobs which have allocated nodes fail during execution.</i></p>

NODEAVAILABILITYPOLICY	
<b>Format</b>	<POLICY>[:<RESOURCE TYPE>] ... where <POLICY> is one of <b>COMBINED</b> , <b>DEDICATED</b> , or <b>UTILIZED</b> and <RESOURCE TYPE> is one of <i>PROC, MEM, SWAP, or DISK</i>
<b>Default</b>	<b>COMBINED</b>
<b>Description</b>	<p>Specifies how available node resources are reported. Moab uses the following calculations to determine the amount of available resources:</p> <p><b>Dedicated</b>(use what Moab has scheduled to be used): Available = Configured - Dedicated</p> <p><b>Utilized</b>(use what the resource manager is reporting is being used): Available = Configured - Utilized</p> <p><b>Combined</b>(use the larger of dedicated and utilized): Available = Configured - (MAX(Dedicated, Utilized))</p> <p>Moab marks a node as busy when it has no available processors, so <b>NODEAVAILABILITYPOLICY</b>, by affecting how many processors are reported as available, also affects node state. See <a href="#">Node Availability Policies</a> for more information.</p>
<b>Example</b>	<pre>NODEAVAILABILITYPOLICY DEDICATED:PROCS COMBINED:MEM</pre> <p><i>Moab will ignore resource utilization information in locating available processors for jobs but will use both dedicated and utilized memory information in determining memory availability.</i></p>

NODEBUSYSTATEDELAYTIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>0:01:00</code> (one minute)
<b>Description</b>	Length of time Moab will assume busy nodes will remain unavailable for scheduling if a system reservation is not explicitly created for the node.
<b>Example</b>	<pre>NODEBUSYSTATEDELAYTIME 0:30:00</pre> <p><i>Moab will assume busy nodes are not available for scheduling for at least <b>30 minutes</b> from the current time. Thus, these nodes will never be allocated to starting jobs. Also, these nodes will only be available for reservations starting more than <b>30 minutes</b> in the future.</i></p>

NODECATCREDLIST	
<b>Format</b>	<code>&lt;LABEL&gt;=&lt;NODECAT&gt;[,&lt;NODECAT&gt;]...[&lt;LABEL&gt;=&lt;NODECAT&gt;[,&lt;NODECAT&gt;]...]</code> where <code>&lt;LABEL&gt;</code> is any string and <code>&lt;NODECAT&gt;</code> is one of the defined node categories.
<b>Default</b>	---
<b>Description</b>	If specified, Moab will generate node category groupings and each iteration will assign usage of matching resources to pseudo-credentials with a name matching the specified label. See the <a href="#">Node Categorization</a> section of the Admin manual for more information.
<b>Example</b>	<pre>NODECATCREDLIST down=BatchFailure,HardwareFailure,NetworkFailure idle=Idle</pre> <p><i>Moab will create a <b>down</b> user, group, account, class, and QoS and will associate <b>BatchFailure</b>, <b>HardwareFailure</b>, and <b>NetworkFailure</b> resources with these credentials. Additionally, Moab will assign all <b>Idle</b> resources to matching <b>idle</b> credentials.</i></p>

NODECFG[X]	
<b>Format</b>	List of space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">ACCESS</a> , <a href="#">CHARGERATE</a> , <a href="#">FEATURES</a> , <a href="#">FLAGS</a> , <a href="#">GRES</a> , <a href="#">MAXJOB</a> , <a href="#">MAXJOBPERUSER</a> , <a href="#">MAXLOAD</a> , <a href="#">MAXPE</a> , <a href="#">NODEINDEX</a> , <a href="#">NODETYPE</a> , <a href="#">OSLIST</a> , <a href="#">PARTITION</a> , <a href="#">POWERPOLICY</a> on page 560, <a href="#">PRIORITY</a> , <a href="#">PRIORITYF</a> , <a href="#">PROCSPEED</a> , <a href="#">RACK</a> , <a href="#">RADISK</a> , <a href="#">SLOT</a> , <a href="#">SPEED</a> , or <a href="#">TRIGGER</a>
<b>Default</b>	---
<b>Description</b>	Specifies node-specific attributes for the node indicated in the array field. See the <a href="#">General Node Administration Overview</a> for more information.
<b>Example</b>	<pre>NODECFG[nodeA] MAXJOB=2 SPEED=1.2</pre> <p><i>Moab will only allow 2 simultaneous jobs to run on node nodeA and will assign a relative machine speed of 1.2 to this node.</i></p>

NODEDOWNSTATEDELAYTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	-1 (never)
<b>Description</b>	Length of time Moab will assume <a href="#">down</a> , <a href="#">drained</a> (offline), or corrupt nodes will remain unavailable for scheduling if a system reservation is not explicitly created for the node. The default specification of "-1" causes Moab to never create job reservations on down nodes. See <a href="#">Node Availability</a> for more information.
<b>Example</b>	<pre>NODEDOWNSTATEDELAYTIME 0:30:00</pre> <p><i>Moab will assume down, drained, and corrupt nodes are not available for scheduling for at least 30 minutes from the current time. Thus, these nodes will never be allocated to starting jobs. Also, these nodes will only be available for reservations starting more than 30 minutes in the future.</i></p>

NODEDOWNTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	The maximum time a previously reported node remains unreported by a resource manager before the node is considered to be in the <a href="#">down</a> state. This can happen if communication with a resource manager or a peer server is lost for more than the specified length of time, or if there is communication with the resource manager but it fails to report the node status.
<b>Example</b>	<pre>NODEDOWNTIME 10:00</pre> <p><i>If Moab loses communication with the resource manager for more than <b>10 minutes</b>, it sets the state of all nodes belonging to that resource manager to <b>DOWN</b>.</i></p>

NODEDRAINSTATEDELAYTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	<b>3:00:00</b> (three hours)
<b>Description</b>	Length of time Moab will assume <a href="#">drained</a> nodes will remain unavailable for scheduling if a system reservation is not explicitly created for the node. Specifying "-1" will cause Moab to never create job reservations on drained nodes. See <a href="#">Node Availability</a> for more information.
<b>Example</b>	<pre>NODEDRAINSTATEDELAYTIME 0:30:00</pre> <p><i>Moab will assume down, drained, and corrupt nodes are not available for scheduling for at least <b>30 minutes</b> from the current time. Thus, these nodes will never be allocated to starting jobs. Also, these nodes will only be available for reservations starting more than <b>30 minutes</b> in the future.</i></p>

NODEFAILURERESERVETIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>0:05:00</code>
<b>Description</b>	Duration of reservation Moab will place on any node in which it detects a failure from the resource manager (0 indicates no reservation will be placed on the node). See <a href="#">Node Availability</a> for more information. See also <a href="#">RMCFG[] NODEFAILURERSVPROFILE</a> . (For related information, see <a href="#">Reservation Policies</a> , <a href="#">DEFERSTARTCOUNT</a> , <a href="#">DEFERTIME</a> , <a href="#">RESERVATIONRETRYTIME</a> , <a href="#">JOBRETRYTIME</a> , and <a href="#">GUARANTEEDPREEMPTION</a> .)
<b>Example</b>	<pre>NODEFAILURERESERVETIME 10:00</pre> <p><i>Moab will reserve failed nodes for 10 minutes.</i></p>

NODEIDFORMAT	
<b>Format</b>	<code>&lt;STRING&gt;</code>
<b>Default</b>	<code>*\$N*</code>
<b>Description</b>	Specifies how a node id can be processed to extract possible node, rack, slot, and cluster index information. The value of the parameter may include the markers <code>\$C</code> (cluster index), <code>\$N</code> (node index), <code>\$R</code> (rack index), or <code>\$S</code> (slot index) separated by <code>*</code> (asterisk - representing any number of non-numeric characters) or other characters to indicate this encoding. See <a href="#">Node Selection</a> for more information on use of node, rack, and slot indices.
<b>Example</b>	<pre>NODEIDFORMAT *\$R*\$S</pre> <p><i>Moab will extract rack and slot information from the cluster node ids (i.e. tg-13s08).</i></p>

NODEIDLEPOWERACTION	
<b>Format</b>	[STANDBY   SUSPEND   SLEEP   HIBERNATE   SHUTDOWN   OFF]
<b>Default</b>	<i>OFF</i>
<b>Description</b>	Specifies what to do with a node that exceeds the NODEIDLEPOWERTHRESHOLD limit.
<b>Example</b>	<pre>PARCFG[ALL] NODEIDLEPOWERACTION STANDBY</pre> <p><i>Nodes that exceed the NODEIDLEPOWERTHRESHOLD limit are placed in standby.</i></p>

NODEIDLEPOWERTHRESHOLD	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>60 seconds</i>
<b>Description</b>	Specifies how long to allow a node to be idle before performing a power action. Increasing the idle duration prevents power on/off thrashing.
<b>Example</b>	<pre>NODEIDLEPOWERTHRESHOLD 300</pre> <p><i>Moab will wait 5 minutes before performing a power action on a node that has become idle.</i></p>

NODEMAXLOAD	
<b>Format</b>	<DOUBLE>
<b>Default</b>	<i>0.0</i>
<b>Description</b>	Specifies that maximum cpu load on an idle or running node. If the node's load reaches or exceeds this value, Moab will mark the node <u>busy</u> .
<b>Example</b>	<pre>NODEMAXLOAD 0.75</pre> <p><i>Moab will adjust the state of all idle and running nodes with a load &gt;= .75 to the state busy.</i></p>

NODEMEMOVERCOMMITFACTOR	
<b>Format</b>	<DOUBLE>
<b>Default</b>	---
<b>Description</b>	The parameter overcommits available and configured memory and swap on a node by the specified factor (for example: mem/swap * factor). Used to show that the node has more mem and swap than it really does. Only works for PBS RM types.
<b>Example</b>	<pre>NODEMEMOVERCOMMITFACTOR .5</pre> <p><i>Moab will overcommit the memory and swap of the node by a factor of 0.5.</i></p>

NODEPOLLFREQUENCY	
<b>Format</b>	<INTEGER>
<b>Default</b>	0 (Poll Always)
<b>Description</b>	Specifies the number of scheduling iterations between scheduler initiated node manager queries. If set to '-2', Moab will never query the node manager daemons. If set to '-1', Moab will only query on the first iteration. <b>Note:</b> this parameter is most often used with OpenPBS and PBSPro. It is not required when using TORQUE, LoadLeveler, LSF, or SGE as the resource managers.
<b>Example</b>	<pre>NODEPOLLFREQUENCY 5</pre> <p><i>Moab will update node manager based information every 5 scheduling iterations.</i></p>

NODESETATTRIBUTE	
<b>Format</b>	<i>FEATURE</i> or <i>VARATTR</i>
<b>Default</b>	---
<b>Description</b>	Specifies the type of node attribute by which node set boundaries will be established. See <a href="#">Node Set Overview</a> .
<b>Example</b>	<pre>NODESETPOLICY    ONEOF NODESETATTRIBUTE FEATURE</pre> <p><i>Moab will create node sets containing nodes with common features.</i></p>

NODESETDELAY	
<b>Format</b>	<i>[[[DD:]HH:]MM:]SS</i>
<b>Default</b>	<i>0:00:00</i>
<b>Description</b>	<p>Causes Moab to attempt to span a job evenly across nodesets unless doing so delays the job beyond the requested <b>NODESETDELAY</b>.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p><b>i</b> Must use with <b>NODESETPLUS</b> on page 1009 set to <i>SPANEVENLY</i>; if you do not want to use <i>SPANEVENLY</i>, use <b>NODESETISOPTIONAL</b> on page 1008 instead of <b>NODESETDELAY</b>.</p> </div>
<b>Example</b>	<pre>NODESETPLUS SPANEVENLY NODESETDELAY    5:00</pre> <p><i>Moab tries to span the job evenly across nodesets unless doing so delays the job by 5 minutes.</i></p>

NODESETISOPTIONAL	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Specifies whether or not Moab will start a job if a requested node set cannot be satisfied. See <a href="#">Node Set Overview</a> .
<b>Example</b>	<pre>NODESETISOPTIONAL TRUE</pre> <p><i>Moab will not block a job from running if its node set cannot be satisfied.</i></p>

NODESETLIST	
<b>Format</b>	<ATTR>[{:}]<ATTR>[...]
<b>Default</b>	---
<b>Description</b>	Specifies the list of node attribute values which will be considered for establishing node sets. See <a href="#">Node Set Overview</a> .
<b>Example</b>	<pre>NODESETPOLICY      ONEOF NODESETATTRIBUTE   FEATURE NODESETLIST        switchA,switchB</pre> <p><i>Moab will allocate nodes to jobs either using only nodes with the <b>switchA</b> feature or using only nodes with the <b>switchB</b> feature.</i></p>

NODESETPLUS	
<b>Format</b>	<i>DELAY</i> or <i>SPANEVENLY</i>
<b>Default</b>	---
<b>Description</b>	<p>Specifies how Moab distributes jobs among nodesets. See <a href="#">Node Set Overview</a>.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> Neither <i>SPANEVENLY</i> nor <i>DELAY</i> values of the <a href="#">NODESETPLUS</a> parameter will work with multi-req jobs or preemption.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 10px;"> <p>NODESETPLUS SPANEVENLY</p> <p><i>Moab attempts to fit all jobs on a single nodeset or to span them evenly across a number of nodesets, unless doing so would delay a job beyond the requested <a href="#">NODESETDELAY</a>.</i></p> </div> <div style="border: 1px dashed #ccc; padding: 10px;"> <p>NODESETPLUS DELAY</p> <p><i>Moab attempts to schedule the job within a nodeset for the configured <a href="#">NODESETDELAY</a>. If Moab cannot find space for the job to start within <a href="#">NODESETDELAY</a> (Moab considers future workload to determine if space will open up in time and might create a future reservation), then Moab schedules the job and ignores the nodeset requirement.</i></p> </div>

NODESETPOLICY	
<b>Format</b>	<i>ANYOF, FIRSTOF, or ONEOF</i>
<b>Default</b>	---
<b>Description</b>	<p>Specifies how nodes will be allocated to the job from the various node set generated. See <a href="#">Node Set Overview</a>.</p>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 10px;"> <p>NODESETPOLICY ONEOF NODESETATTRIBUTE NETWORK</p> </div> <div style="border: 1px dashed #ccc; padding: 10px;"> <p><i>Moab will create node sets containing nodes with common network interfaces.</i></p> </div>

NODESETPRIORITYTYPE	
<b>Format</b>	one of <i>AFFINITY</i> , <i>BESTFIT</i> , <i>WORSTFIT</i> , or <i>MINLOSS</i>
<b>Default</b>	<i>MINLOSS</i>
<b>Description</b>	Specifies how resource sets will be selected when more than one feasible resource can be found. See <a href="#">Node Set Overview</a> .
<b>Example</b>	<pre>NODESETPRIORITYTYPE BESTFIT NODESETATTRIBUTE    PROC SPEED</pre> <p><i>Moab will select the resource set that most closely matches the set of resources requested.</i></p>

NODESYNCTIME	
<b>Format</b>	<i>[[[DD:]HH:]MM:]SS</i>
<b>Default</b>	<i>00:10:00</i>
<b>Description</b>	Specifies the length of time after which Moab will sync up a node's expected state with an unexpected reported state. <b>IMPORTANT Note:</b> Moab will not start new jobs on a node with an expected state which does not match the state reported by the resource manager.
<b>Example</b>	<pre>NODESYNCTIME 1:00:00</pre>

NODETOJOBATTRMAP	
<b>Format</b>	Comma delimited list of node features
<b>Default</b>	---
<b>Description</b>	Job requesting the listed node features will be assigned a corresponding job attribute. These job attributes can be used to enable <a href="#">reservation access</a> , adjust <a href="#">job priority</a> or enable other capabilities.
<b>Example</b>	<pre>NODETOJOBATTRMAP fast, big</pre> <p><i>Jobs requesting node feature <i>fast</i> or <i>big</i> will be assigned a corresponding job attribute.</i></p>

NODEUNTRACKEDRESDELAYTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0:00:00
<b>Description</b>	<p>Length of time Moab will assume untracked generic resources will remain unavailable for scheduling if a system reservation is not explicitly created for the node.</p> <p>If <b>NODEUNTRACKEDRESDELAYTIME</b> is enabled and there is an untracked resource preventing a job from running, then the job remains in the idle queue instead of being deferred.</p>
<b>Example</b>	<div style="border: 1px dashed gray; padding: 10px;"> <p>NODEUNTRACKEDRESDELAYTIME 0:30:00</p> <p><i>Moab will assume untracked generic resources are not available for scheduling for at least 30 minutes from the current time. Thus, these nodes will never be allocated to starting jobs. Also, these nodes will only be available for reservations starting more than 30 minutes in the future.</i></p> </div>

NODEVMFEATURECHECKTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0:10:00
<b>Description</b>	<p>The length of time between each Moab check on node and VM features. If a running VM requires a feature but the resource manager is no longer reporting that feature on the VM's host node, Moab migrates the VM to a node that has the feature. If no other node has that feature, no migration occurs.</p>
<b>Example</b>	<div style="border: 1px dashed gray; padding: 10px;"> <p>NODEVMFEATURECHECKTIME 10:00</p> <p><i>Moab checks node and VM features every 10 minutes.</i></p> </div>

NODEVMREQATTRCHECKTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	0:10:00

NODEVMREQATTRCHECKTIME	
<b>Description</b>	The length of time between each Moab check on a VM's requested node attributes. If a running VM requires node attributes but the resource manager is no longer reporting requested attributes on the VM's host node, Moab migrates the VM to a node that has the requested attributes. If no other node has the requested attributes, no migration occurs.
<b>Example</b>	<pre>NODEVMREQATTRCHECKTIME 10:00</pre> <p><i>Moab checks requested node attributes of a node running a VM every 10 minutes.</i></p>

NODEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight which will be applied to a job's requested node count before this value is added to the job's cumulative priority. <b>Note:</b> this weight currently only applies when a nodecount is specified by the user job. If the job only specifies tasks or processors, no node factor will be applied to the job's total priority. This will be rectified in future versions.
<b>Example</b>	<pre>NODEWEIGHT 1000</pre>

NOLOCALUSERENV	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If <i>TRUE</i> , specifies that a user's UserID, GroupID, and HomeDirectory are available on the Moab server host.
<b>Example</b>	<pre>NOLOCALUSERENV TRUE</pre>

NOJOBHOLDNORESOURCES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If <i>TRUE</i> , Moab does not place a hold on jobs that don't have feasible resources. For example, suppose there are 20 processors available for ClassA and 50 processors for the entire system. If a job requests 21 or more processors from ClassA, or 51 or more processors from the entire system, Moab idles the job (instead of putting a hold on it) until the resources become available.
<b>Example</b>	<pre>NOJOBHOLDNORESOURCES TRUE</pre>

NOTIFICATIONPROGRAM	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the name of the program to handle all notification call-outs.
<b>Example</b>	<pre>NOTIFICATIONPROGRAM tools/notifyme.pl</pre>

NOWAITPREEMPTION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	Generally when a job is trying to preempt another, it just waits for the original jobs that it chose to preempt to end. If this parameter is on, the preemptor will continue trying to preempt jobs until it can get in.
<b>Example</b>	<pre>NOWAITPREEMPTION TRUE</pre>

OSCREDLLOOKUP	
<b>Format</b>	NEVER
<b>Default</b>	---
<b>Description</b>	<p>Disables all Moab OS credential lookups, including UID, GID, user to group mappings, and any other OS specific information.</p> <p>Setting OSCREDLLOOKUP by itself does not allow job submission; additional configuration is required. When submitting jobs from user accounts that do not exist on the head node (where Moab HPC Suite – Basic Edition and TORQUE are running), you must also set the <a href="#">PROXYJOB SUBMISSION</a> flag in addition to specifying configuration settings in the resource manager configuration file. See the example that follows for information on required resource manager settings.</p>
<b>Example</b>	<pre>OSCREDLLOOKUP NEVER RMCFG[] FLAGS=PROXYJOB SUBMISSION</pre> <p>To allow job submission, in the TORQUE configuration file (<a href="#">torque.cfg</a>):</p> <pre>VALIDATEPATH FALSE</pre> <p>Run the following <a href="#">qmgr</a> directive:</p> <pre>set server disable_server_id_check = True</pre> <p>Restart both Moab HPC Suite – Basic Edition and pbs_server.</p>

PARALLOCATIONPOLICY	
<b>Format</b>	One of <i>BestFit</i> , <i>BestFitP</i> , <i>FirstStart</i> , <i>LoadBalance</i> , <i>LoadBalanceP</i> , <i>Random</i> , or <i>RoundRobin</i>
<b>Default</b>	<i>FirstStart</i>
<b>Description</b>	Specifies the approach to use to allocate resources when more than one eligible partition can be found. See <a href="#">Grid Scheduling Policies</a> for more information.
<b>Example</b>	<pre>PARALLOCATIONPOLICY LOADBALANCE</pre> <p><i>New jobs will be started on the most lightly allocated partition.</i></p>

PARCFG	
<b>Format</b>	<a href="#">NODEPOWEROFFDURATION</a> , <a href="#">NODEPOWERONDURATION</a> , <a href="#">NODEALLOCATIONPOLICY</a> or one or more key-value pairs as described in the <a href="#">Partition Overview</a>
<b>Default</b>	---
<b>Description</b>	Specifies the attributes, policies, and constraints for the given partition.
<b>Example</b>	<pre>PARCFG[oldcluster] MAX.WCLIMIT=12:00:00</pre> <p><i>Moab will not allow jobs to run on the oldcluster partition which has a wallclock limit in excess of 12 hours.</i></p>

PBSACCOUNTINGDIR	
<b>Format</b>	<PATH>
<b>Default</b>	---
<b>Description</b>	When specified, Moab will write out job events in standard PBS/ <a href="#">TORQUE tracejob</a> format to the specified directory using the standard PBS/TORQUE log file naming convention.
<b>Example</b>	<pre>PBSACCOUNTINGDIR /var/spool/torque/sched_logs/</pre> <p><i>Job events will be written to the specified directory (can be consumed by PBS's tracejob command).</i></p>

PERPARTITIONSCHEDULING	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>By default Moab's scheduling routine schedules each job on each partition using the following algorithm:</p> <pre> <i>prioritize</i> <i>foreach (job)</i>     <i>find the partition on which that job should run</i>     <i>schedule job</i> </pre> <p>In this model, a job's priority is the same on each partition as it uses a single global priority. Because a job's priority is the same on every partition, Moab prioritizes the queue once and then schedules the prioritized queue across all partitions.</p> <p>When PERPARTITIONSCHEDULING TRUE is set, the following algorithm is used:</p> <pre> <i>foreach (partition)</i>     <i>prioritize</i>     <i>foreach (job)</i>         <i>schedule job</i> </pre> <p>In this case, each partition may have a unique priority configuration and Moab will re-prioritize the jobs for each partition on the system. Each job is prioritized and scheduled on each partition. See <a href="#">PARCFG on page 1015</a> for more information. Also, note that Moab will order the partitions as they are discovered in the moab.cfg file. Partitions should be explicitly ordered via PARCFG in the moab.cfg file.</p>
<b>Example</b>	<pre> PERPARTITIONSCHEDULING TRUE PARCFG [p1] CONFIGFILE=/opt/moab/etc/p1.cfg PARCFG [p2] CONFIGFILE=/opt/moab/etc/p2.cfg </pre> <p><i>Rather than prioritizing the job queue once, Moab prioritizes the job queue for each partition, p1 and p2 respectively, and schedules each partition in turn using the policies located in their respective configuration files. (See <a href="#">Per-Partition Settings on page 497</a> for more information).</i></p>

PEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the coefficient to be multiplied by a job's PE (processor equivalent) priority factor.
<b>Example</b>	<pre>RESWEIGHT 10 PEWEIGHT 100</pre> <p><i>Each job's priority will be increased by 10 * 100 * its PE factor.</i></p>

PREEMTPOLICY	
<b>Format</b>	one of the following: <a href="#">CANCEL</a> , <a href="#">REQUEUE</a> , <a href="#">SUSPEND</a> , or <a href="#">CHECKPOINT</a>
<b>Default</b>	<i>REQUEUE</i>
<b>Description</b>	Specifies how preemptable jobs will be <a href="#">preempted</a> . <b>Note:</b> If this policy is set to <a href="#">REQUEUE</a> , preemptible jobs should be marked as <i>RESTARTABLE</i> . If this policy is set to <a href="#">SUSPEND</a> , preemptible jobs should be marked as <i>SUSPENDABLE</i> . <b>Note:</b> Moab uses preemption escalation to preempt resources if the specified preemption facility is not applicable. This means if the policy is set to <i>SUSPEND</i> and the job is not <i>SUSPENDABLE</i> , Moab may attempt to requeue or even cancel the job.
<b>Example</b>	<pre>PREEMTPOLICY CHECKPOINT</pre> <p><i>Jobs that are to be preempted will be checkpointed and restarted at a later time.</i></p>

PREEMTPRIOJOBSELECTWEIGHT	
<b>Format</b>	<DOUBLE>
<b>Default</b>	256.0
<b>Description</b>	<p>Determines which jobs to preempt based on size or priority. The higher the value, the more emphasis is placed on the priority of the job, causing the lower priority jobs to be preempted first. The lower the value, the more emphasis is placed on the size of the job, causing the smaller jobs to be preempted first. If set to 0, job priority will be ignored, job size will take precedence and the smallest jobs will be preempted.</p> <p>The special setting of -1 places the emphasis solely on resource utilization. This means that jobs will be preempted in a manner that keeps the resource utilization at the highest level, regardless of job priority or size.</p>
<b>Example</b>	<pre>PREEMTPRIOJOBSELECTWEIGHT 220.5</pre>

PREEMPTRIMEWEIGHT	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0
<b>Description</b>	<p>If set to anything other than 0, a job's remaining time is added into the calculation of which jobs will be preempted. If a positive weight is specified, jobs with a longer remaining time are favored. If a negative weight is specified, jobs with a shorter remaining time are favored.</p>
<b>Example</b>	<pre>PREEMPTRIMEWEIGHT 1.5</pre>

PREEMPTSEARCHDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>unlimited</i>
<b>Description</b>	Specifies how many preemptible jobs will be evaluated as potential targets for serial job preemptors. See <a href="#">Preemption Overview</a> for more information.
<b>Example</b>	<pre>PREEMPTSEARCHDEPTH 8</pre> <p><i>Serial job preemptors will only consider the first 8 feasible preemptee jobs when determining the best action to take.</i></p>

PRIORITYTARGETDURATION	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	Specifies the <i>ideal</i> job duration which will maximize the value of the <a href="#">WALLTIMEWEIGHT</a> priority factor. If specified, this factor will be calculated as the distance from the ideal. Consequently, in most cases, the associated subcomponent weight should be set to a negative value.
<b>Example</b>	<pre>WALLTIMEWEIGHT -2500 PRIORITYTARGETDURATION 1:00:00</pre>

PRIORITYTARGETPROCCOUNT	
<b>Format</b>	<INTEGER>{+ - %}
<b>Default</b>	---
<b>Description</b>	Specifies the ideal job requested proc count which will maximize the value of the <a href="#">PROCWEIGHT</a> priority factor. If specified, this factor will be calculated as the distance from the ideal (proc count - ideal = coefficient of PROCWEIGHT). Consequently, in most cases, the associated subcomponent weight should be set to a negative value.
<b>Example</b>	<pre>PROCWEIGHT -1000 PRIORITYTARGETPROCCOUNT 64</pre>

PROCWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the coefficient to be multiplied by a job's requested processor count priority factor.
<b>Example</b>	<code>PROCWEIGHT 2500</code>

PROFILECOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	600
<b>Description</b>	Specifies the number of statistical profiles to maintain. PROFILECOUNT must be set high enough that at least one day of statistics is maintained. The statistics time window can be determined by measuring PROFILEDURATION * PROFILECOUNT. If <a href="#">PROFILEDURATION</a> is one hour then PROFILECOUNT must be at least 24 so 24 hours worth of statistics are maintained. If PROFILEDURATION is 30:00 then PROFILECOUNT must be set to at least 48. If PROFILECOUNT is not high enough for at least one day of statistics, Moab adjusts it automatically.
<b>Example</b>	<code>PROFILECOUNT 300</code>

PROFILEDURATION	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	00:30:00
<b>Description</b>	Specifies the duration of each statistical profile. The duration cannot be more than 24 hours, and any specified duration must be a factor of 24. For example, factors of 1/4, 1/2, 1, 2, 3, 4, 6, 8, 12, and 24 are acceptable durations.
<b>Example</b>	<code>PROFILEDURATION 24:00:00</code>

PURGETIME	
<b>Format</b>	<code>[[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>0</code>
<b>Description</b>	The amount of time Moab will keep a job or node record for an object no longer reported by the resource manager. Useful when using a resource manager which 'drops' information about a node or job due to internal failures. Note: This parameter is superseded by <a href="#">JOBPURGETIME on page 985</a> .
<b>Example</b>	<pre>PURGETIME 00:05:00</pre> <p><i>Moab will maintain a job or node record for 5 minutes after the last update regarding that object received from the resource manager.</i></p>

PUSHCACHETOWEBSERVICE	
<b>Format</b>	<code>&lt;BOOLEAN&gt;</code>
<b>Default</b>	<code>FALSE</code>
<b>Description</b>	Specifies whether or not you want to send cache objects (nodes, jobs, services, etc.) to Moab Web Services.
<b>Example</b>	<pre>PUSHCACHETOWEBSERVICE TRUE</pre>

QOSCFG[<QOSID>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">General Credential Flags</a> , <b>PRIORITY</b> , <a href="#">ENABLEPROFILING</a> , <b>FSTARGET</b> , <a href="#">JOBPRIOACCRUALPOLICY</a> , <a href="#">JOBPRIOEXCEPTIONS</a> , <a href="#">MEMBERULIST</a> , <b>QTWEIGHT</b> , <b>QTTARGET</b> , <b>XFWEIGHT</b> , <b>XFTARGET</b> , <a href="#">PREEMPTMINTIME</a> , <a href="#">PREEMPTMAXTIME</a> , <a href="#">PREEMPTQTTHRESHOLD</a> , <a href="#">PREEMPTXFTHRESHOLD</a> , <a href="#">PREEMPTTEES</a> , <a href="#">RSVQTTHRESHOLD</a> , <a href="#">RSVXFTHRESHOLD</a> , <a href="#">ACLBLTHRESHOLD</a> , <a href="#">ACLQTTHRESHOLD</a> , <a href="#">ACLXFTHRESHOLD</a> , <b>PLIST</b> , <a href="#">QFLAGS</a> , or a <a href="#">usage limit</a> .
<b>Default</b>	---
<b>Description</b>	Specifies QOS specific attributes. See the <a href="#">flag overview</a> for a description of legal flag values. See the <a href="#">QOS Overview</a> section for further details.
<b>Example</b>	<pre>QOSCFG[commercial] PRIORITY=1000 MAXJOB=4 MAXPROC=80</pre> <p><i>Moab will increase the priority of jobs using QOS commercial, and will allow up to 4 simultaneous QOS commercial jobs with up to 80 total allocated processors.</i></p>

QOSDEFAULTORDER	
<b>Format</b>	Comma-delimited list of QOS names.
<b>Default</b>	---
<b>Description</b>	Sets a global QOS default order for all QOS's which overrides any specific default QOS. If the order is defined as b, a, c and a user has access to c, a and submits a job without requesting a specific QOS, the job is assigned a as the default QOS.
<b>Example</b>	<pre>QOSDEFAULTORDER b,a,c</pre> <p><i>If the job does not have a QOS specified, it is assigned a QOS from the QOSDEFAULTORDER list (if the user has access to one of them).</i></p>

QOSISOPTIONAL	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	An entity's default QOS will be the first QOS specified in the QLIST parameter. When this parameter is set to <i>TRUE</i> the default QOS for the associated credential (user, account, class, etc.) will not be automatically set to the first QOS specified in the QLIST.
<b>Example</b>	<pre>QOSISOPTIONAL TRUE USERCFG[bob] QLIST=high, low</pre> <p><i>Moab will set the QOSList for user bob to <b>high</b> and <b>low</b> but will not set the QDEF. Should bob decide to submit to a particular QOS he will have to do so manually.</i></p>

QOSREJECTPOLICY	
<b>Format</b>	One or more of <i>CANCEL</i> , <i>HOLD</i> , <i>IGNORE</i> , or <i>MAIL</i>
<b>Default</b>	<i>HOLD</i> ( <i>IGNORE</i> for SLURM users)
<b>Description</b>	Specifies the action to take when Moab determines that a job cannot access a requested <a href="#">QoS</a> . <i>CANCEL</i> issues a call to the resource manager to cancel the job. <i>HOLD</i> places a <i>batch</i> hold on the job preventing the job from being further evaluated until released by an administrator. ( <b>Note:</b> Administrators can dynamically alter job attributes and possibly <i>fix</i> the job with <a href="#">mjobctl -m</a> .) With <i>IGNORE</i> , Moab will ignore the QoS request and schedule the job using the default QoS for that job. <i>MAIL</i> will send email to both the admin and the user when QoS request violations are detected. Most combinations of attributes may be specified; however, if both <i>MAIL</i> and <i>IGNORE</i> are specified, Moab will not implement <i>MAIL</i> . Similarly, while <i>CANCEL</i> and <i>HOLD</i> are mutually exclusive, <i>CANCEL</i> will supersede <i>HOLD</i> if both are specified. (see <a href="#">JOBREJECTPOLICY</a> ).
<b>Example</b>	<pre>QOSREJECTPOLICY MAIL, CANCEL</pre>

QOSWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the weight to be applied to the qos priority of each job (see <a href="#">Credential (CRED) Factor</a> ).
<b>Example</b>	<pre>QOSWEIGHT 10</pre>

QUEUETIMECAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum allowed absolute pre-weighted queuetime priority factor.
<b>Example</b>	<pre>QUEUETIMECAP 10000 QUEUETIMEWEIGHT 10</pre> <p><i>A job that has been queued for 40 minutes will have its queuetime priority factor calculated as 'Priority = QUEUETIMEWEIGHT * MIN(10000,40)'.</i></p>

QUEUETIMEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies multiplier applied to a job's queue time (in minutes) to determine the job's queuetime priority factor.
<b>Example</b>	<pre>QUEUETIMEWEIGHT 20</pre> <p><i>A job that has been queued for 4:20:00 will have a queuetime priority factor of 20 * 260.</i></p>

REALTIMEDBOBJECTS	
<b>Format</b>	Comma-delimited list of one or more of the following: <i>JOB</i> , <i>NODE</i> , <i>RSV</i> (reservation), <i>TRIG</i> (trigger), <i>VC</i> (virtual container). You can also specify <i>ALL</i> or <i>NONE</i> .
<b>Default</b>	<i>ALL</i>
<b>Description</b>	Specifies which objects Moab will store in the unixodbc database.
<b>Example</b>	<pre>REALTIMEDBOBJECTS JOB,RSV,TRIG</pre> <p><i>Moab stores jobs, reservations, and triggers in the uxodbc database. It will no longer record real time information about nodes and VCs.</i></p>

RECORDEVENTLIST	
<b>Format</b>	One or more comma (',') or plus ('+') separated events of <a href="#">GEVENT</a> , <a href="#">ALLSCHEDCOMMAND</a> , <a href="#">AMCREATE</a> , <a href="#">AMDELETE</a> , <a href="#">AMEND</a> , <a href="#">AMPAUSE</a> , <a href="#">AMQUOTE</a> , <a href="#">AMRESUME</a> , <a href="#">AMSTART</a> , <a href="#">AMUPDATE</a> , <a href="#">JOBCANCEL</a> , <a href="#">JOBCHECKPOINT</a> , <a href="#">JOBEND</a> , <a href="#">JOBFAILURE</a> , <a href="#">JOBMIGRATE</a> , <a href="#">JOBMODIFY</a> , <a href="#">JOBPREEMPT</a> , <a href="#">JOBREJECT</a> , <a href="#">JOBRESUME</a> , <a href="#">JOBSTART</a> , <a href="#">JOBSUBMIT</a> , <a href="#">NODEDOWN</a> , <a href="#">NODEFAILURE</a> , <a href="#">NODEUP</a> , <a href="#">QOSVIOLATION</a> , <a href="#">RMDOWN</a> , <a href="#">RMPOLLEND</a> , <a href="#">RMPOLLSTART</a> , <a href="#">RMUP</a> , <a href="#">RSVCANCEL</a> , <a href="#">RSVCREATE</a> , <a href="#">RSVEND</a> , <a href="#">RSVMODIFY</a> , <a href="#">RSVSTART</a> , <a href="#">SCHEDCOMMAND</a> , <a href="#">SCHEDCYCLEEND</a> , <a href="#">SCHEDCYCLESTART</a> , <a href="#">SCHEDPAUSE</a> , <a href="#">SCHEDSTART</a> , <a href="#">SCHEDSTOP</a> , <a href="#">VMCREATE</a> , <a href="#">VMDESTROY</a> , <a href="#">VMMIGRATE</a> , <a href="#">VMPOWEROFF</a> , <a href="#">VMPOWERON</a> , or <a href="#">ALL</a>
<b>Default</b>	<a href="#">JOBSTART</a> , <a href="#">JOBCANCEL</a> , <a href="#">JOBEND</a> , <a href="#">JOBFAILURE</a> , <a href="#">SCHEDPAUSE</a> , <a href="#">SCHEDSTART</a> , <a href="#">SCHEDSTOP</a> , <a href="#">TRIGEND</a> , <a href="#">TRIGFAILURE</a> , <a href="#">TRIGSTART</a>
<b>Description</b>	Specifies which events should be recorded in the appropriate event file found in Moab's <code>stats/</code> directory. These events are recorded for both local and remotely staged jobs. (See <a href="#">Event Log Overview</a> ) <b>Note:</b> If a plus character is included in the list, the specified events will be added to the default list; otherwise, the specified list will replace the default list.
<b>Example</b>	<pre>RECORDEVENTLIST JOBSTART, JOBCANCEL, JOBEND</pre> <p><i>When a local and/or remote job starts, is canceled, or ends, the respective event will be recorded.</i></p>

REJECTDOSSCRIPTS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<a href="#">TRUE</a>
<b>Description</b>	Moab rejects DOS-formatted scripts submitted with the <code>msub</code> command. This is useful if you use SLURM as your resource manager, since it does not handle DOS scripts well. For <a href="#">REJECTDOSSCRIPTS</a> to work correctly, <a href="#">FILTERCMDFILE</a> on page 957 must be <a href="#">FALSE</a> . Otherwise, Moab modifies the script instead of rejecting it, leading to job errors.
<b>Example</b>	<pre>REJECTDOSSCRIPTS FALSE</pre> <p><i>Moab does not reject DOS-formatted scripts submitted with <code>msub</code>.</i></p>

REJECTINFEASIBLEJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	If zero feasible nodes are found for a job among the currently available nodes on the cluster, the scheduler rejects the job. See <a href="#">JOBREJECTPOLICY</a> for more information.
<b>Example</b>	<pre>REJECTINFEASIBLEJOBS TRUE JOBREJECTPOLICY MAIL, CANCEL</pre> <p><i>Any job with zero feasible nodes for execution will be rejected.</i></p>

REJECTNEGPRIOJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	If enabled, the scheduler will refuse to start any job with a negative priority. See <a href="#">Job Priority Overview</a> and <a href="#">ENABLENEGJOBPRIORITY</a> for more information.
<b>Example</b>	<pre>ENABLENEGJOBPRIORITY TRUE REJECTNEGPRIOJOBS TRUE</pre> <p><i>Any job with a priority less than zero will be rejected.</i></p>

REMAPCLASS	
<b>Format</b>	<ClassID>
<b>Default</b>	---
<b>Description</b>	<p>Specifies which class/queue will be remapped based on the processors, nodes, and node features requested and the resource limits of each <a href="#">class</a>. See <a href="#">Remap Class Overview</a> for more information.</p> <div style="border: 1px solid #0070C0; border-radius: 5px; padding: 5px; background-color: #D9E1F2;"> <p> In order to use <b>REMAPCLASS</b>, you must specify a <b>DEFAULTCLASS</b>.</p> </div>
<b>Example</b>	<pre>RMCFG[internal]  DEFAULTCLASS=batch REMAPCLASS      batch CLASSCFG[small]  MAX.PROC=2 CLASSCFG[medium] MAX.PROC=16 CLASSCFG[large]  MAX.PROC=1024</pre> <div style="border: 1px dashed #000; padding: 5px; margin-top: 10px;"> <p><i>Class <b>batch</b> will be remapped based on the number of processors requested.</i></p> </div>

REMAPCLASSLIST	
<b>Format</b>	Comma delimited list of class names
<b>Default</b>	---
<b>Description</b>	<p>Specifies the order in which classes will be searched when attempting to <a href="#">remap</a> a class. Only classes included in the list will be searched and Moab will select the first class with matches. <b>Note:</b> If no <b>REMAPCLASSLIST</b> is specified, Moab will search all classes and will search them in the order they are discovered. See <a href="#">Remap Class Overview</a> for more information.</p>
<b>Example</b>	<pre>RMCFG[internal]  DEFAULTCLASS=batch REMAPCLASS      batch REMAPCLASSLIST  short,medium,long</pre> <div style="border: 1px dashed #000; padding: 5px; margin-top: 10px;"> <p><i>Class <b>batch</b> will be re-mapped to one of the listed classes.</i></p> </div>

REMOTEFAILTRANSIENT	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Only applicable to Moab configurations with multiple resource managers able to run jobs (such as in a grid environment). When Moab attempts to migrate a job to one of these resource managers, a remote failure may occur. For example, a destination peer in a grid that has an error accepting a job results in a remote error, and the job is rejected. <b>REMOTEFAILTRANSIENT</b> controls how Moab reacts to remote errors. By default, Moab considers such an error permanent and does not try to migrate the same job to that resource manager again. If <b>REMOTEFAILTRANSIENT</b> is set to <i>TRUE</i> , then Moab considers such an error as transient and will not exclude the erring resource manager in future migration attempts.
<b>Example</b>	<pre>REMOTEFAILTRANSIENT  TRUE</pre>

REMOVETRIGOUTPUTFILES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	When Moab launches external trigger actions, the standard output and error of those trigger actions are redirected to files located in Moab's spool directory. By default, these files are cleaned every 24 hours. (Files older than 24 hours are removed.) If, however, you wish to have Moab immediately remove the spool files after they are no longer needed, set <code>RemoveTrigOutputFiles</code> to <i>TRUE</i> .
<b>Example</b>	<pre>REMOVETRIGOUTPUTFILES  TRUE</pre>

RESCAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum allowed absolute pre-weighted job resource priority factor.
<b>Example</b>	<pre>RESCAP 1000</pre> <p><i>The total resource priority factor component of a job will be bound by +/- 1000</i></p>

RESERVATIONDEPTH[X]	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the number of priority reservations which are allowed in the associated reservation bucket. <b>Note:</b> The array index, X, is the bucket label and can be any string up to 64 characters. This label should be synchronized with the <a href="#">RESERVATIONQOSLIST</a> parameter. See <a href="#">Reservation Policies</a> .
<b>Example</b>	<pre>RESERVATIONDEPTH[bigmem] 4 RESERVATIONQOSLIST[bigmem] special,fast,joshua</pre> <p><i>Jobs with QOS's of special, fast, or joshua can have a cumulative total of up to 4 priority reservations.</i></p>

RESERVATIONPOLICY	
<b>Format</b>	One of the following: <i>CURRENTHIGHEST, HIGHEST, NEVER</i>
<b>Default</b>	<i>CURRENTHIGHEST</i>
<b>Description</b>	Specifies how Moab reservations will be handled. (See also <a href="#">RESERVATIONDEPTH</a> ) See <a href="#">Reservation Policies</a> .
<b>Example</b>	<pre>RESERVATIONPOLICY          CURRENTHIGHEST RESERVATIONDEPTH [DEFAULT] 2</pre> <p><i>Moab will maintain reservations for only the 2 currently highest priority jobs.</i></p>

RESERVATIONQOSLIST[X]	
<b>Format</b>	One or more QOS values or <i>[ALL]</i>
<b>Default</b>	<i>[ALL]</i>
<b>Description</b>	Specifies which QOS credentials have access to the associated reservation bucket. <b>Note:</b> The array index, X, is the bucket label and can be any string up to 64 characters. This label should be synchronized with the <a href="#">RESERVATIONDEPTH</a> parameter. See <a href="#">Reservation Policies</a> .
<b>Example</b>	<pre>RESERVATIONDEPTH [big]    4 RESERVATIONQOSLIST [big] hi, low, med</pre> <p><i>Jobs with QOS's of hi, low, or med can have a cumulative total of up to 4 priority reservations.</i></p>

RESERVATIONRETRYTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	60 seconds
<b>Description</b>	Period of time Moab will continue to attempt to allocate resources to start a job after the time resources should be made available. This parameter takes into account resource manager node state race conditions, nodes with residual high load, network glitches, etc. (For related information, see <a href="#">Reservation Policies</a> , <a href="#">DEFERSTARTCOUNT</a> , <a href="#">DEFERTIME</a> , <a href="#">NODEFAILURERESERVETIME</a> , <a href="#">JOBRETRYTIME</a> , and <a href="#">GUARANTEEDPREEMPTION</a> .)
<b>Example</b>	<pre>RESERVATIONRETRYTIME 00:05:00</pre> <p><i>Moab will try for up to 5 minutes to maintain immediate reservations if the reservations are blocked due to node state, network, or batch system based race conditions.</i></p>

RESOURCELIMITMULTIPLIER[<PARID>]	
<b>Format</b>	<p>&lt;RESOURCE&gt;:&lt;MULTIPLIER&gt;[,...]</p> <p>Where &lt;RESOURCE&gt; is one of the following:  <b>NODE, PROC, JOBPROC, MEM, JOBMEM, SWAP, DISK, or WALLTIME</b></p>
<b>Default</b>	1.0
<b>Description</b>	If set to less than one, then the hard limit will be the specified limit and the soft limit will be the specified limit multiplied by the multiplier. If set to a value greater than one, then the specified limit will be the soft limit and the hard limit will be the specified limit multiplied by the multiplier. See <a href="#">Usage-based Limits</a> .
<b>Example</b>	<pre>RESOURCELIMITMULTIPLIER PROC:1.1, MEM:2.0</pre> <p><i>Sets hard limit for <b>PROC</b> at 1.1 times the <b>PROC</b> soft limit, and the hard limit of <b>MEM</b> to 2.0 times the <b>MEM</b> soft limit.</i></p>

RESOURCELIMITPOLICY	
<b>Format</b>	<p>&lt;RESOURCE&gt;:[&lt;SPOLICY&gt;,&lt;HPOLICY&gt; : [&lt;SACTION&gt;,&lt;HACTION&gt;] [[:&lt;SVIOLATIONTIME&gt;,&lt;HVIOLATIONTIME&gt;]..</p> <p>Where <b>RESOURCE</b> is one of <i>CPUTIME, DISK, JOBMEM, JOBPROC, MEM, MINJOBPROC, NETWORK, PROC, SWAP, or WALLTIME</i></p> <p>where <b>*POLICY</b> is one of <i>ALWAYS, EXTENDEDVIOLATION, or BLOCKEDWORKLOADONLY</i></p> <p>and where <b>*ACTION</b> is one of <i>CANCEL, CHECKPOINT, NOTIFY, REQUEUE, SIGNAL, or SUSPEND.</i></p>
<b>Default</b>	No limit enforcement.
<b>Description</b>	Specifies how the scheduler should handle jobs which utilize more resources than they request. See <a href="#">Usage-based Limits</a> .
<b>Example</b>	<pre>RESOURCELIMITPOLICY MEM:ALWAYS, BLOCKEDWORKLOADONLY:REQUEUE, CANCEL</pre> <p><i>Moab will cancel all jobs which exceed their requested memory limits.</i></p>

RESTARTINTERVAL	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	Causes Moab daemon to recycle/restart when the given interval of time has transpired.
<b>Example</b>	<pre>RESTARTINTERVAL 20:00:00</pre> <p><i>Moab daemon will automatically restart every 20 hours.</i></p>

RESOURCEQUERYDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	3
<b>Description</b>	Maximum number of options which will be returned in response to an <a href="#">mshow -a</a> resource query.
<b>Example</b>	<pre>RESOURCEQUERYDEPTH 1</pre> <p><i>The mshow -a command will return at most 1 valid collection of resources.</i></p>

RESWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	All resource priority components are multiplied by this value before being added to the total job priority. See <a href="#">Job Prioritization</a> .
<b>Example</b>	<pre>RESWEIGHT 5 MEMWEIGHT 10 PROCWEIGHT 100 SWAPWEIGHT 0 RESCAP 2000</pre> <p><i>The job priority resource factor will be calculated as <math>\text{MIN}(2000, 5 * (10 * \text{JobMemory} + 100 * \text{JobProc}))</math>.</i></p>

RMCFG	
<b>Format</b>	One or more key-value pairs as described in the <a href="#">Resource Manager Configuration Overview</a>
<b>Default</b>	---
<b>Description</b>	Specifies the interface and policy configuration for the scheduler-resource manager interface. Described in detail in the <a href="#">Resource Manager Configuration Overview</a> .
<b>Example</b>	<pre>RMCFG[TORQUE3] TYPE=PBS</pre> <p><i>The PBS server will be used for resource management.</i></p>

RMMSGIGNORE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether or not Moab should adjust node state based on generic resource manager failure messages. See <a href="#">Compute node health check on page 2677</a> for more info.
<b>Example</b>	<pre>RMMSGIGNORE TRUE</pre> <p><i>Moab will load and report resource manager failure messages but will not adjust node state as a result of them.</i></p>

RMPOLLINTERVAL	
<b>Format</b>	[<MINPOLLTIME>,<MAXPOLLTIME> where poll time is specified as [[[DD:]HH:]MM:]SS
<b>Default</b>	0,30
<b>Description</b>	<p>Specifies the interval between RM polls. The poll interval will be no less than MINPOLLTIME and no more than MAXPOLLTIME. If you specify a single value, Moab interprets the value as the MAXPOLLTIME with a MINPOLLTIME of 0.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> If you use TORQUE as your resource manager, prevent communication errors by giving <a href="#">tcp_timeout</a> on page 2783 at least twice the value of the Moab <b>RMPOLLINTERVAL</b>.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 10px;"> <pre>RMPOLLINTERVAL 30,45</pre> </div> <p><i>Moab will refresh its resource manager information between a minimum of 30 seconds and a maximum of 45 seconds. Note: This parameter specifies the default global poll interval for all resource managers.</i></p>

RMRETRYTIMECAP	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	1:00:00
<b>Description</b>	<p>Moab attempts to contact RMs that are in state 'corrupt' (not down). If the attempt is unsuccessful, Moab tries again later. If the second attempt is unsuccessful, Moab increases the gap (the gap grows exponentially) between communication attempts. <b>RMRETRYTIMECAP</b> puts a cap on the length between connection attempts.</p>
<b>Example</b>	<div style="border: 1px dashed #ccc; padding: 10px; margin-bottom: 10px;"> <pre>RMRETRYTIMECAP 24:00:00</pre> </div> <p><i>Moab stops increasing the gap between connection attempts once the retry gap reaches 24 hours.</i></p>

RSVLIMITPOLICY	
<b>Format</b>	<i>HARD</i> or <i>SOFT</i>
<b>Default</b>	---
<b>Description</b>	Specifies what limits should be enforced when creating reservations.
<b>Example</b>	<pre>RSVLIMITPOLICY HARD</pre> <p><i>Moab will limit reservation creation based on the <b>HARD</b> limits configured.</i></p>

RSVNODEALLOCATIONPOLICY	
<b>Format</b>	One of the following: <a href="#">FIRSTAVAILABLE</a> , <a href="#">LASTAVAILABLE</a> , <a href="#">MINRESOURCE</a> , <a href="#">CPULOAD</a> , <a href="#">LOCAL</a> , <a href="#">CONTIGUOUS</a> , <a href="#">MAXBALANCE</a> , or <a href="#">PRIORITY</a>
<b>Default</b>	<i>LASTAVAILABLE</i>
<b>Description</b>	Specifies how Moab should allocate available resources to reservations.
<b>Example</b>	<pre>RSVNODEALLOCATIONPOLICY MINRESOURCE</pre> <p><i>Moab will apply the node allocation policy <b>MINRESOURCE</b> to all reservations by default.</i></p>

RSVNODEALLOCATIONPRIORITYF	
<b>Format</b>	User specified algorithm
<b>Default</b>	---
<b>Description</b>	When <a href="#">RSVNODEALLOCATIONPOLICY</a> is set to <i>PRIORITY</i> , this parameter allows you to specify your own priority algorithm. The priority functions available are the same as the <a href="#">node priority functions</a> .
<b>Example</b>	<pre>RSVNODEALLOCATIONPOLICY PRIORITY RSVNODEALLOCATIONPRIORITYF 'SPEED + .01 * AMEM - 10 * JOBCOUNT'</pre>

RSVPROFILE[X]	
<b>Format</b>	<p>One or more of the following:</p> <p><i>Allowed:</i></p> <p><a href="#">TRIGGERACL</a> (<a href="#">ACCOUNTLIST</a>, <a href="#">CLASSLIST</a>, <a href="#">GROUPLIST</a>, <a href="#">MAXTIME</a>, <a href="#">QOSLIST</a>, <a href="#">USERLIST</a>)</p> <p>HostExp ( <a href="#">HOSTLIST</a>)</p> <p>Features (<a href="#">NODEFEATURES</a>)</p> <p><a href="#">FLAGS</a></p> <p><a href="#">TASKCOUNT</a></p> <p><a href="#">RSVACCESSLIST</a></p> <p><b>Note:</b> Lists of more than one ACL value cannot be whitespace delimited. Such lists must be delimited with a comma, pipe, or colon.</p> <p><i>Not allowed:</i></p> <p><a href="#">ACCESS</a></p> <p><a href="#">CHARGEACCOUNT</a></p> <p><a href="#">DAYS</a></p> <p><a href="#">DEPTH</a></p> <p><a href="#">ENDTIME</a></p> <p><a href="#">OWNER</a></p> <p><a href="#">PARTITION</a></p> <p><a href="#">PERIOD</a></p> <p><b>PRIORITY</b></p> <p><a href="#">RESOURCES</a></p> <p><a href="#">STARTTIME</a></p> <p><a href="#">TPN</a></p>
<b>Default</b>	---
<b>Description</b>	Specifies attributes of a reservation profile using syntax similar to that for specifying a standing reservation. See <a href="#">Using Reservation Profiles</a> for details.
<b>Example</b>	<pre style="border: 1px dashed gray; padding: 10px;"> RSVPROFILE[fast] USERLIST=john,steve RSVPROFILE[fast] QOSLIST=high,low RSVPROFILE[fast] TRIGGER=ETYPE=start,OFFSET=5:00,ATYPE=exec,ACTION="/opt/moab/rp.pl"                 </pre> <p style="border: 1px dashed gray; padding: 5px; margin-top: 10px;"><i>Moab will create a reservation profile including trigger and ACL information.</i></p>

RSVSEARCHALGO	
<b>Format</b>	<i>LONG</i> or <i>WIDE</i>
<b>Default</b>	<i>NONE</i>
<b>Description</b>	<p>When Moab is determining when and where a job can run, it either searches for the most resources (<i>WIDE</i>) or the longest range of resources (<i>LONG</i>). In almost all cases, searching for the longest range is ideal and returns the soonest starttime. In some rare cases, however, a particular job may need to search for the most resources. In those cases sites can configure this parameter to prevent the starvation of large jobs that fail to hold onto their reservation starttimes. See the <a href="#">WIDERSVSEARCHALGO</a> job flag.</p> <p>If this parameter is not set, it will be displayed in <code>mschedctl -l</code> as <i>NONE</i> but the algorithm that is used will be <i>LONG</i>.</p>
<b>Example</b>	<pre>RSVSEARCHALGO WIDE</pre>

SCHEDCFG	
<b>Format</b>	List of zero or more space delimited <code>&lt;ATTR&gt;=&lt;VALUE&gt;</code> pairs where <code>&lt;ATTR&gt;</code> is one of the following: <a href="#">FBSERVER</a> , <a href="#">FLAGS</a> , <a href="#">MAXJOBID</a> , <a href="#">MAXRECORDEDCJOBID</a> , <a href="#">MINJOBID</a> , <a href="#">HTTPSERVERPORT</a> , <a href="#">MODE</a> , <a href="#">RECOVERYACTION</a> , <a href="#">SERVER</a> , or <a href="#">TRIGGER</a>
<b>Default</b>	---
<b>Description</b>	<p>Specifies scheduler policy and interface configuration.</p> <div style="border: 1px solid #0056b3; padding: 5px; margin-top: 10px;"> <p><b>i</b> The <b>SERVER</b> attribute can also be set using the environment variable <b>\$MOABSERVER</b>. Using this variable allows you to quickly change the Moab server that client commands will connect to.</p> <pre>&gt; export MOABSERVER=cluster2:12221</pre> </div>
<b>Example</b>	<pre>SCHEDCFG[zylem3] SERVER=geronimo.scc.com:3422 MODE=NORMAL</pre> <p><i>Moab will execute in <b>NORMAL</b> mode on the host <b>geronimo.scc.com</b>.</i></p>

SERVERHOST	
<b>Format</b>	<HOSTNAME>
<b>Default</b>	---
<b>Description</b>	<b>Deprecated.</b> Hostname of machine on which Moab will run. See <a href="#">SCHEDCFG</a> for replacement parameter.
<b>Example</b>	<pre>SERVERHOST geronimo.scc.edu</pre> <p><i>Moab will execute on the host <a href="#">geronimo.scc.edu</a>.</i></p>

SERVERMODE	
<b>Format</b>	One of the following: <a href="#">INTERACTIVE</a> , <a href="#">MONITOR</a> , <i>NORMAL</i> , <i>SIMULATION</i> , or <i>SLAVE</i>
<b>Default</b>	<i>NORMAL</i>
<b>Description</b>	<b>Deprecated.</b> Specifies how Moab interacts with the outside world. See <a href="#">SCHEDCFG</a> for replacement parameter.
<b>Example</b>	<pre>SERVERMODE SIMULATION</pre>

SERVERNAME	
<b>Format</b>	<STRING>
<b>Default</b>	<SERVERHOST>
<b>Description</b>	Specifies the name the scheduler will use to refer to itself in communication with peer daemons. See <a href="#">SCHEDCFG</a> for replacement parameter.
<b>Example</b>	<pre>SERVERNAME moabA</pre>

SERVERPORT	
<b>Format</b>	<INTEGER> (range: 1-64000)
<b>Default</b>	40559
<b>Description</b>	Port on which moab will open its user interface socket. See <a href="#">SCHEDCFG</a> for replacement parameter.
<b>Example</b>	<pre>SERVERPORT 30003</pre> <p><i>Moab will listen for client socket connections on port 30003.</i></p>

SERVERSUBMITFILTER	
<b>Format</b>	<PATH>
<b>Default</b>	---
<b>Description</b>	Specifies the location of a global job submit filter script. When you configure a global job submit filter, Moab executes it on the head node and uses it to filter every job submission it receives. See <a href="#">Server-based submit filter on page 305</a> for more information about job submit filters.
<b>Example</b>	<pre>SERVERSUBMITFILTER /opt/moab/scripts/globalfilter.pl</pre> <p><i>Moab uses /opt/moab/scripts/globalfilter.pl to filter every job submitted to Moab.</i></p>

SERVICEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the service component weight associated with the service factors. See <a href="#">Service (SERV) Factor</a> for more information.
<b>Example</b>	<pre>SERVICEWEIGHT 2</pre>

SHOWMIGRATEDJOBSASIDLE	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	By default, migrated jobs in the grid will show as blocked. This is to prevent jobs from counting against the idle policies of multiple clusters rather than just the cluster to which the job was migrated.
<b>Example</b>	<pre>SHOWMIGRATEDJOBSASIDLE TRUE</pre> <p><i>When set to <b>TRUE</b>, migrated jobs will show as idle and will count against the idle policies of the cluster showing the job as migrated.</i></p>

SIMAUTOSHUTDOWN	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>TRUE</i>
<b>Description</b>	If <i>TRUE</i> , the scheduler will end simulations when the active queue and idle queue become empty.
<b>Example</b>	<pre>SIMAUTOSHUTDOWN TRUE</pre> <p><i>The simulation will end as soon as there are no jobs running and no idle jobs which could run.</i></p>

SIMINITIALQUEUEDEPTH	
<b>Format</b>	<INTEGER>
<b>Default</b>	16
<b>Description</b>	Specifies how many jobs the simulator will initially place in the idle job queue (see Simulation Overview).
<b>Example</b>	<pre>SCHEDCFG[sim1]          MODE=SIMULATION SIMINITIALQUEUEDEPTH    64 SIMJOBSSUBMISSIONPOLICY CONSTANTJOBDEPTH</pre> <p><i>Moab will initially place 64 idle jobs in the queue and, because of the specified queue policy, will attempt to maintain this many jobs in the idle queue throughout the duration of the simulation.</i></p>

SIMJOBSSUBMISSIONPOLICY	
<b>Format</b>	One of the following: <i>NORMAL</i> , <i>CONSTANTJOBDEPTH</i> , <i>CONSTANTPSDEPTH</i> , or <a href="#">REPLAY</a>
<b>Default</b>	<i>CONSTANTJOBDEPTH</i>
<b>Description</b>	Specifies how the simulator will submit new jobs into the idle queue. <i>NORMAL</i> mode causes jobs to be submitted at the time recorded in the workload trace file, <i>CONSTANTJOBDEPTH</i> and <i>CONSTANTPSDEPTH</i> attempt to maintain an idle queue of <a href="#">SIMINITIALQUEUEDEPTH</a> jobs and proc-seconds respectively. <i>REPLAY</i> will force jobs to execute at the exactly the time specified in the simulation job trace file. This mode is most often used to generate detailed profile statistics for analysis in <a href="#">Moab Cluster Manager</a> (see Simulation Overview).
<b>Example</b>	<pre>SIMJOBSSUBMISSIONPOLICY NORMAL</pre> <p><i>Moab will submit jobs with the relative time distribution specified in the workload trace file.</i></p>

SIMPURGEBLOCKEDJOBS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Specifies whether Moab should remove jobs which can never execute (see Simulation Overview).
<b>Example</b>	<pre>SIMPURGEBLOCKEDJOBS FALSE</pre>

SIMMRANDOMDELAY	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the random delay added to the RM command base delay accumulated when making any resource manager call in simulation mode.
<b>Example</b>	<pre>SIMMRANDOMDELAY 5</pre> <p><i>Moab will add a random delay of between 0 and 5 seconds to the simulated time delay of all RM calls.</i></p>

SIMSTARTTIME	
<b>Format</b>	[HH]:MM[:SS]][[_MO[/DD[/YY]]]
<b>Default</b>	---
<b>Description</b>	Specifies the time when the simulation starts.
<b>Example</b>	<pre>SIMSTARTTIME 00:00:00_01/01/00</pre> <p><i>Moab will set its clock to <b>January 1, 2000 at 12:00:00</b> in the morning before starting the simulation</i></p>

SIMSTOPTIME	
<b>Format</b>	<code>[HH[:MM[:SS]]][_MO[/DD[/YY]]]</code>
<b>Default</b>	---
<b>Description</b>	Specifies the time when the simulation should pause.
<b>Example</b>	<pre>SIMSTOPTIME 00:00:00_01/01/04</pre> <p><i>Moab will stop scheduling when its internal simulation time reaches January 1, 2004.</i></p>

SIMWORKLOADTRACEFILE	
<b>Format</b>	<code>&lt;STRING&gt;</code>
<b>Default</b>	<code>Traces/workload.trace</code>
<b>Description</b>	Specifies the file from which moab will obtain job information when running in simulation mode. Moab will attempt to locate the file relative to <code>&lt;MOABHOMEDIR&gt;</code> unless specified as an absolute path. See Simulation Overview and <a href="#">Workload Accounting Records</a> .
<b>Example</b>	<pre>SIMWORKLOADTRACEFILE traces/jobs.2</pre> <p><i>Moab will obtain job traces when running in simulation mode from the <code>&lt;MOABHOMEDIR&gt;/traces/jobs.2</code> file.</i></p>

SPOOLDIR	
<b>Format</b>	<code>&lt;STRING&gt;</code>
<b>Default</b>	---
<b>Description</b>	Specifies the directory for temporary spool files created by Moab while submitting a job to the RM.
<b>Example</b>	<pre>SPOOLDIR /tmp/moab/spool</pre>

SPOOLDIRKEEPTIME	
<b>Format</b>	<INTEGER> (seconds) or [[[DD:]HH:]MM:]SS
<b>Default</b>	---
<b>Description</b>	Specifies the interval to delete spool files and other temporary files that have been left in the spool directory.
<b>Example</b>	<pre>SPOOLDIRKEEPTIME 4:00:00</pre>

SPVIOLATIONWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight to be applied to a job which violates <a href="#">soft</a> usage limit policies (see <a href="#">Service (SERVICE) Component on page 384</a> ).
<b>Example</b>	<pre>SPVIOLATIONWEIGHT 5000</pre>

SRCFG[X]	
<b>Format</b>	<p>One or more of the following &lt;ATTR&gt;=&lt;VALUE&gt; pairs  <a href="#">ACCESS</a>, <a href="#">ACCOUNTLIST</a>, <a href="#">CHARGE</a> on page 467, <a href="#">CHARGEACCOUNT</a>, <a href="#">CHARGEUSER</a>, <a href="#">CLASSLIST</a>,  <a href="#">CLUSTERLIST</a>, <a href="#">COMMENT</a>, <a href="#">DAYS</a>, <a href="#">DEPTH</a>, <a href="#">DISABLE</a>, <a href="#">ENDTIME</a>, <a href="#">FLAGS</a>, <a href="#">GROUPLIST</a>, <a href="#">HOSTLIST</a>,  <a href="#">JOBATTRLIST</a>, <a href="#">MAXTIME</a>, <a href="#">NODEFEATURES</a>, <a href="#">OWNER</a>, <a href="#">PARTITION</a>, <a href="#">PERIOD</a>, <a href="#">PRIORITY</a>, <a href="#">QOSLIST</a>,  <a href="#">REQUIREDTPN</a>, <a href="#">RESOURCES</a>, <a href="#">ROLLBACKOFFSET</a>, <a href="#">RSVACCESSLIST</a>, <a href="#">RSVGROUP</a>, <a href="#">STARTTIME</a>,  <a href="#">TASKCOUNT</a>, <a href="#">TIMELIMIT</a>, <a href="#">TPN</a>, <a href="#">TRIGGER</a>, or <a href="#">USERLIST</a>  <b>Note:</b>HOSTLIST and ACL list values must be comma delimited. For example:  <b>HOSTLIST=nodeA,nodeB</b></p>
<b>Default</b>	---
<b>Description</b>	Specifies attributes of a standing reservation. See <a href="#">Managing Reservations</a> for details.
<b>Example</b>	<pre>SRCFG[fast] STARTTIME=9:00:00 ENDTIME=15:00:00 SRCFG[fast] HOSTLIST=node0[1-4]\$ SRCFG[fast] QOSLIST=high,low</pre> <p><i>Moab will create a standing reservation running from 9:00 AM to 3:00 PM on nodes 1 through 4 accessible by jobs with QOS high or low.</i></p>

STARTCOUNTCAP	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the max weighted value allowed from the startcount subfactor when determining a job's priority (see <a href="#">Priority Factors</a> for more information).
<b>Example</b>	<pre>STARTCOUNTWEIGHT 5000 STARTCOUNTCAP 30000</pre>

STARTCOUNTWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight to be applied to a job's startcount when determining a job's priority (see <a href="#">Priority Factors</a> for more information).
<b>Example</b>	<pre>STARTCOUNTWEIGHT 5000</pre>

STATDIR	
<b>Format</b>	<STRING>
<b>Default</b>	stats
<b>Description</b>	Specifies the directory in which Moab statistics will be maintained.
<b>Example</b>	<pre>STATDIR /var/adm/moab/stats</pre>

STATPROCMAX	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	<p>Specifies the maximum number of processors requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via <code>mschedctl -m</code> or <code>changeparam</code> while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-bottom: 10px;"> <pre>STATPROCMAX      256 STATPROCSTEPCOUNT 4 STATPROCSTEPsize 4</pre> </div> <p><i>Each matrix output will display data in rows for jobs requesting between 4 and 256 processors.</i></p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p><b>i</b> A <b>NONE</b> in services will still allow users to run <a href="#">showq</a> and <a href="#">checkjob</a> on their own jobs.</p> </div>

STATPROCMIN	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	<p>Specifies the minimum number of processors requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;">STATPROCMIN      4 STATPROCSTEPCOUNT 4 STATPROCSTEPsize  4</pre> <p style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"><i>Each matrix output will display data in rows for jobs requesting between 4 and 256 processors.</i></p> <div style="border: 1px solid #0070C0; padding: 5px;"> <p><b>i</b> A NONE in services will still allow users to run <a href="#">showq</a> and <a href="#">checkjob</a> on their own jobs.</p> </div>

STATPROCSTEPCOUNT	
<b>Format</b>	<INTEGER>
<b>Default</b>	5
<b>Description</b>	<p>Specifies the number of rows of processors requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;">STATPROCMIN      4 STATPROCSTEPCOUNT 4 STATPROCSTEPsize  4</pre> <p style="border: 1px dashed #ccc; padding: 5px;"><i>Each matrix output will display data in rows for jobs requesting between 4 and 256 processors.</i></p>

STATPROCSTEPSIZE	
<b>Format</b>	<INTEGER>
<b>Default</b>	4
<b>Description</b>	<p>Specifies the processor count multiplier for rows of processors requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre>STATPROCMIN      4 STATPROCSTEPCOUNT 4 STATPROCSTEPSIZE 4</pre> <p><i>Each matrix output will display data in rows for jobs requesting between 4 and 256 processors.</i></p>

STATTIMEMAX	
<b>Format</b>	[[DD:]HH:]MM:]SS
<b>Default</b>	00:15:00
<b>Description</b>	<p>Specifies the maximum amount of time requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre>STATTIMEMAX      02:08:00 STATTIMESTEPCOUNT 4 STATTIMESTEPSIZE 4</pre> <p><i>Each matrix output will display data in columns for jobs requesting between 2 and 128 minutes.</i></p>

STATTIMEMIN	
<b>Format</b>	<code>[[DD:]HH:]MM:]SS</code>
<b>Default</b>	<code>00:15:00</code>
<b>Description</b>	<p>Specifies the minimum amount of time requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre>STATTIMEMIN      00:02:00 STATTIMESTEPCOUNT  4 STATTIMESTEPSIZE   4</pre> <div style="border: 1px dashed #000; padding: 5px; margin-top: 10px;"> <p><i>Each matrix output will display data in columns for jobs requesting between 2 and 128 minutes.</i></p> </div>

STATTIMESTEPCOUNT	
<b>Format</b>	<code>&lt;INTEGER&gt;</code>
<b>Default</b>	<code>6</code>
<b>Description</b>	<p>Specifies the number of columns of time requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre>STATTIMEMIN      00:02:00 STATTIMESTEPCOUNT  4 STATTIMESTEPSIZE   4</pre> <div style="border: 1px dashed #000; padding: 5px; margin-top: 10px;"> <p><i>Each matrix output will display data in columns for jobs requesting between 2 and 128 minutes.</i></p> </div>

STATTIMESTEPSIZE	
<b>Format</b>	<INTEGER>
<b>Default</b>	4
<b>Description</b>	<p>Specifies the time multiplier for columns of time requested by jobs to be displayed in matrix outputs (as displayed by the <a href="#">showstats -f</a> command).</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> It is recommended that you not change any parameters via 'mschedctl -m' or 'changeparam' while Moab is running. Changing any of the parameters invalidates all past data and will start the collection over.</p> </div>
<b>Example</b>	<pre>STATTIMEMIN      00:02:00 STATTIMESTEPCOUNT 4 STATTIMESTEPSIZE 4</pre> <p><i>Each matrix output will display data in columns for jobs requesting between 2 and 128 minutes.</i></p>

STOPITERATION	
<b>Format</b>	<INTEGER>
<b>Default</b>	-1 (don't stop)
<b>Description</b>	Specifies which scheduling iteration Moab will stop and wait for a command to resume scheduling.
<b>Example</b>	<pre>STOPITERATION 10</pre> <p><i>Moab should stop after iteration 10 of scheduling and wait for administrator commands.</i></p>

STOREJOB SUBMISSION	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	<p>When set to <i>TRUE</i>, specifies that Moab will save a job's submit arguments and script to \$MOABHOMEDIR/stats/jobarchive/jobNumber.</p> <p>If you use TORQUE as your resource manager, you can configure it to store completed job information, and it will store the same information returned by the <code>qstat -f</code> command. For more information, see <a href="#">Job logging on page 2664</a> in the TORQUE documentation.</p>
<b>Example</b>	<pre>STOREJOB SUBMISSION TRUE</pre>

STRICTPROTOCOLCHECK	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>Specifies how Moab reacts to differences in XML protocols when communicating with other Moab peers. If set to <i>TRUE</i>, Moab will reject any communication that does not strictly conform to the expected protocol. If set to <i>FALSE</i> (the default), Moab will not reject XML that has extra or unknown attributes.</p>
<b>Example</b>	<pre>STRICTPROTOCOLCHECK TRUE</pre> <p><i>Moab will reject any XML communication that does not strictly conform to the expected protocol definition.</i></p>

SUBMITENVFILELOCATION	
<b>Format</b>	<i>FILE</i> or <i>PIPE</i>
<b>Default</b>	---
<b>Description</b>	<p>If set to <i>FILE</i>, these behaviors are expected:</p> <ul style="list-style-type: none"> <li>• The environment file is owned by a user with 600 permissions.</li> <li>• Moab writes the environment variables ('\0' delimited) to a random file in Moab's spool directory.</li> <li>• Moab adds the <code>--export-file=&lt;path_to_file&gt;</code> on the sbatch command line.</li> <li>• Moab deletes the file after the job completes.</li> </ul> <p>If set to <i>PIPE</i>, these behaviors are expected:</p> <ul style="list-style-type: none"> <li>• Moab creates a pipe and passes the read end of the pipe's file descriptor to sbatch.</li> <li>• Moab's parent process writes the environment ('\0' delimited) into the write end of the pipe.</li> </ul> <p>Adaptive Computing recommends that you configure this parameter for a more secure environment.</p>
<b>Example</b>	<code>SUBMITENVFILELOCATION PIPE</code>

SUBMITFILTER	
<b>Format</b>	< <i>STRING</i> >
<b>Default</b>	---
<b>Description</b>	Specifies the directory of a given <a href="#">submit filter script</a> .
<b>Example</b>	<code>SUBMITFILTER /home/submitfilter/filter.pl</code>

SUBMITHOSTS	
<b>Format</b>	space delimited list of host names
<b>Default</b>	---
<b>Description</b>	If specified, <b>SUBMITHOSTS</b> specifies an explicit list of hosts where jobs can be submitted.
<b>Example</b>	<code>SUBMITHOSTS hostA hostB</code>

SUSPENDRESOURCES[<PARID>]	
<b>Format</b>	<RESOURCE>[,...] Where <RESOURCE> is one of the following: <i>NODE, PROC, MEM, SWAP, DISK</i>
<b>Default</b>	---
<b>Description</b>	List of resources to dedicate while a job is suspended (available in Moab version 4.5.1 and higher).
<b>Example</b>	<pre>SUSPENDRESOURCES [base] MEM, SWAP, DISK</pre> <p><i>While a job is suspended in partition base, the memory, swap and disk for that job will remain dedicated to the job.</i></p>

SYSCFG	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <b>PRIORITY, FSTARGET, QLIST, QDEF, PLIST, FLAGS</b> , or a <a href="#">fairness policy</a> specification.
<b>Default</b>	---
<b>Description</b>	Specifies system-wide default attributes. See the <a href="#">Attribute/Flag Overview</a> for more information.
<b>Example</b>	<pre>SYSCFG PLIST=Partition1 QDEF=highprio</pre> <p><i>By default, all jobs will have access to partition <b>Partition1</b> and will use the QOS <b>highprio</b>.</i></p>

SWAPWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight assigned to the virtual memory request of a job.
<b>Example</b>	<pre>SWAPWEIGHT 10</pre>

SYSTEMMAXPROCPERJOB	
<b>Format</b>	<INTEGER>
<b>Default</b>	-1 (NO LIMIT)
<b>Description</b>	Specifies the maximum number of processors that can be requested by any single job.
<b>Example</b>	<pre>SYSTEMMAXPROCPERJOB 256</pre> <p><i>Moab will reject jobs requesting more than 256 processors.</i></p>

SYSTEMMAXPROCSECONDPERJOB	
<b>Format</b>	<INTEGER>
<b>Default</b>	-1 (NO LIMIT)
<b>Description</b>	Specifies the maximum number of proc-seconds that can be requested by any single job.
<b>Example</b>	<pre>SYSTEMMAXJOBPROCSECOND 86400</pre> <p><i>Moab will reject jobs requesting more than 86400 procs seconds. i.e, 64 processors * 30 minutes will be rejected, while a 2 processor * 12 hour job will be allowed to run.</i></p>

SYSTEMMAXJOBWALLTIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	-1 (NO LIMIT)
<b>Description</b>	Specifies the maximum amount of wallclock time that can be requested by any single job.
<b>Example</b>	<pre>SYSTEMMAXJOBWALLTIME 1:00:00:00</pre> <p><i>Moab will reject jobs requesting more than 1 day of walltime.</i></p>

TARGETQUEUEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to the time remaining until the queue time is reached.
<b>Example</b>	<code>TARGETQUEUEWEIGHT 10</code>

TARGETWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the weight to be applied to a job's queue time and expansion factor target components (see <a href="#">Job Prioritization</a> ).
<b>Example</b>	<code>TARGETWEIGHT 1000</code>

TARGETXFACTORWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to the distance to the target expansion factor.
<b>Example</b>	<code>TARGETXFACTORWEIGHT 10</code>

TASKDISTRIBUTIONPOLICY	
<b>Format</b>	One of <i>DEFAULT</i> , <i>PACK</i> , <i>RR</i> (round-robin)
<b>Default</b>	---
<b>Description</b>	Specifies how job tasks should be mapped to allocated resources. <i>DEFAULT</i> allows the resource manager to determine how the tasks are placed on the nodes. When <i>PACK</i> is used, a node is filled up with tasks before the next node is used. When <i>RR</i> is used, tasks are cycled through nodes, one task at a time, until there are no more tasks. See <a href="#">Task Distribution Overview</a> for more information.
<b>Example</b>	<pre>TASKDISTRIBUTIONPOLICY DEFAULT</pre> <p><i>Moab should use standard task distribution algorithms.</i></p>

THREADPOOLSIZE	
<b>Format</b>	<INTEGER>
<b>Default</b>	<i>2X number of core processors</i> (MAX: 64)
<b>Description</b>	Governs the number of threads used when processing job scheduling. Scalability and performance may improve with multi-threading; to throttle, limit the number of threads used.
<b>Example</b>	<pre>THREADPOOLSIZE 10</pre>

TOOLSDIR	
<b>Format</b>	<STRING>
<b>Default</b>	<i>tools</i>
<b>Description</b>	Specifies the directory in which Moab tools will be maintained (commonly used in conjunction with <a href="#">Native Resource Managers</a> , and <a href="#">Triggers</a> ).
<b>Example</b>	<pre>TOOLSDIR /var/adm/moab/tools</pre>

TRAPFUNCTION	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the functions to be trapped.
<b>Example</b>	<pre>TRAPFUNCTION UpdateNodeUtilization GetNodeSResTime</pre>

TRAPJOB	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the jobs to be trapped.
<b>Example</b>	<pre>TRAPJOB pros23.0023.0</pre>

TRAPNODE	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the nodes to be trapped.
<b>Example</b>	<pre>TRAPNODE node001 node004 node005</pre>

TRAPRES	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the reservations to be trapped.
<b>Example</b>	<pre>TRAPRES interactive.0.1</pre>

TRIGCHECKTIME	
<b>Format</b>	<INTEGER> (milliseconds)
<b>Default</b>	2000
<b>Description</b>	Each scheduling iteration, Moab will have a period of time where it handles commands and other UI requests. This time period is controlled by <a href="#">RMPOLLINTERVAL</a> . During this time period, known as the UI phase, Moab will periodically evaluate triggers. Usually this only takes a fraction of a second, but if the number of triggers are large it could take up substantially more time (up to several seconds). While Moab is evaluating triggers, it doesn't respond to UI commands. This makes Moab feel sluggish and unresponsive. To remedy this, use the parameter <b>TRIGCHECKTIME</b> . This parameter tells Moab to only spend up to X milliseconds processing triggers during the UI phase. After X milliseconds has gone by, Moab will pause the evaluating of triggers, handle any pending UI events, and then restart the trigger evaluations where it last left off.
<b>Example</b>	<pre>TRIGCHECKTIME 4000</pre>

TRIGEVALLIMIT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Each scheduling iteration, Moab will have a period of time where it handles commands and other UI requests. This time period is controlled by <a href="#">RMPOLLINTERVAL</a> . During this time period, known as the UI phase, Moab will periodically evaluate triggers. The number of times Moab evaluates all triggers in the system is controlled by the <b>TRIGEVALLIMIT</b> parameter. By default, this is set to 1. This means that Moab will evaluate all triggers at most once during the UI phase. Moab will not leave the UI phase and start other scheduling tasks until ALL triggers are evaluated at least one time. If TrigEvalLimit is set to 5, then Moab will wait until all triggers are evaluated five times.
<b>Example</b>	<pre>TRIGEVALLIMIT 3</pre>

UJOBWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Weight assigned by jobs per user. -1 will reduce priority by number of active jobs owned by user.
<b>Example</b>	<pre>UJOBWEIGHT 10</pre>

UMASK	
<b>Format</b>	<INTEGER>
<b>Default</b>	0022 (octal) (produces 0644 permissions)
<b>Description</b>	Specifies the file permission mask to use when creating new fairshare, stats, and event files. See the <code>umask</code> man page for more details.
<b>Example</b>	<pre>UMASK 0127</pre> <i>Create statistics and event files which are 'read-write' by owner and 'read' by group only.</i>

UNSUPPORTEDDEPENDENCIES	
<b>Format</b>	Comma delimited string
<b>Default</b>	---
<b>Description</b>	Specifies <a href="#">dependencies</a> that are not supported and should not be accepted by job submissions. A maximum of 30 dependencies is supported.
<b>Example</b>	<pre># moab.cfg UNSUPPORTEDDEPENDENCIES before,beforeok,beforenotok,on  &gt; msub -l depend=before:105 cmd.sh ERROR: cannot submit job - error in extension string</pre>

UPROCWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Weight assigned by processors per user. -1 will reduce priority by number of active procs owned by user.
<b>Example</b>	<pre>UPROCWEIGHT 10</pre>

USAGECONSUMEDWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to per job processor second consumption.
<b>Example</b>	<pre>USAGECONSUMEDWEIGHT 10</pre>

USAGEEXECUTIONTIMEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight assigned to the total job execution time (measured in seconds since job start). See <a href="#">Preemption Overview</a> .
<b>Example</b>	<code>USAGEEXECUTIONTIMEWEIGHT 10</code>

USAGEPERCENTWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to total requested resources consumed.
<b>Example</b>	<code>USAGEPERCENTWEIGHT 5</code>

USAGEREMAININGWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight assigned to remaining usage.
<b>Example</b>	<code>USAGEREMAININGWEIGHT 10</code>

USAGEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the weight assigned to the percent and total job usage subfactors.
<b>Example</b>	<pre>USAGEWEIGHT 100</pre>

USEANYPARTITIONPRIO	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	<p>The FSTREE data from the first feasible FSTREE will be used when determining a job's start priority, rather than having no FSTREE data considered.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p><b>i</b> Do not set <b>USEANYPARTITIONPRIO</b> if you use per-partition scheduling. Doing so causes to schedule jobs to the first partition listed, even if nodes from another partition will be available sooner.</p> </div>
<b>Example</b>	<pre>USEANYPARTITIONPRIO TRUE</pre>

USECPRSVNODELIST	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Specifies whether Moab should use the checkpointed reservation node list when rebuilding reservations on startup. If this is not used then Moab will use the reservation's specified host expression during rebuilding.
<b>Example</b>	<pre>USECPRSVNODELIST FALSE</pre>

USEDATABASE	
<b>Format</b>	INTERNAL
<b>Default</b>	-
<b>Description</b>	Specifies whether Moab should store profile statistics, checkpoint information, and event information in an integrated database. See <a href="#">Layout of Scheduler Components with Integrated Database Enabled</a> for more information.
<b>Example</b>	<pre>USEDATABASE INTERNAL</pre>

USEJOBREGEX	
<b>Format</b>	BOOLEAN
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether <code>mjobctl</code> supports regular expressions.
<b>Example</b>	<pre>USEJOBREGEX TRUE  [user@linux]\$ mjobctl -c 8[1-3]  job '81' cancelled job '82' cancelled job '83' cancelled</pre>

USEMOABCTIME	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	<p>When Moab finds new jobs on the resource manager, it creates a job inside of Moab for each job in the resource manager. By default, when Moab creates a new job, it uses the time the job was submitted to the resource manager to calculate how long the job has been in the queue (Moab processing time - job creation in resource manager), which is then used in determining the job's priority.</p> <p>In a system where more jobs are submitted to a resource manager than Moab can handle in one iteration, there is the possibility of jobs running out of order. For example, two jobs are both submitted at time 5. The first submitted job is processed first at time 6. So the first job's effective queue duration would be 1 (6-5). On the next iteration, the second job is processed at time 8. So the second job's effective queue duration would be 3 (8-5), indicating that it has been in the queue longer than the other job. Since the later job has a higher effective queue duration it will get a higher priority and could be scheduled to run before earlier submitted jobs.</p> <p>Setting <b>USEMOABCTIME</b> to <i>TRUE</i> tells Moab to use the creation time of the job in Moab rather than the creation time in the resource manager. This corrects the possible problem of having later submitted jobs having higher priorities and starting before earlier submitted jobs.</p>
<b>Example</b>	<code>USEMOABCTIME TRUE</code>

USEMOABJOBID	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	Specifies whether to use the Moab job ID, or the resource manager's job ID.
<b>Example</b>	<code>USEMOABJOBID TRUE</code>

USERCFG[<USERID>]	
<b>Format</b>	List of zero or more space delimited <ATTR>=<VALUE> pairs where <ATTR> is one of the following: <a href="#">General Credential Flags</a> , <b>CDEF</b> , <b>DEFAULT.TPN</b> , <b>DEFAULT.WCLIMIT</b> , <b>EMAILADDRESS</b> , <a href="#">ENABLEPROFILING</a> , <b>FSCAP</b> , <b>FSTARGET</b> , <a href="#">JOBFLAGS</a> , <b>MAX.WCLIMIT</b> , <b>QLIST</b> , <b>QDEF</b> , <b>NOEMAIL</b> , <b>OVERRUN</b> , <b>PLIST</b> , <a href="#">PRIORITY</a> , or a <a href="#">usage limit</a> .
<b>Default</b>	---
<b>Description</b>	Specifies user specific attributes. For general user attribute information, See the <a href="#">Credential Overview</a> . For a description of legal flag values, see <a href="#">flag overview</a> .
<b>Example</b>	<pre>USERCFG[john] MAXJOB=50 QDEF=highprio USERCFG[john] EMAILADDRESS=john@company.com</pre> <p><i>Up to 50 jobs submitted under the user ID john will be allowed to execute simultaneously and will be assigned the QOS highprio.</i></p>

USERPRIOCAP	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Specifies the priority cap to be applied to the user specified job priority factor. Under Moab, only negative user priorities may be specified. See <a href="#">Credential (Service) Factor</a> .
<b>Example</b>	<pre>USERPRIOWEIGHT 10 USERPRIOCAP -10000</pre>

USERPRIOWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight to be applied to the user specified job priority. Under Moab, only negative user priorities may be specified. If this weight is set, users may reduce the priority of some of their jobs to allow other jobs to run earlier. See <a href="#">Credential (Service) Factor</a> and <a href="#">User Selectable Prioritization</a> .
<b>Example</b>	<pre>USERPRIOWEIGHT 10</pre>

USERWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the weight to be applied to the user <a href="#">priority</a> of each job. See <a href="#">Credential (CRED) Factor</a> .
<b>Example</b>	<pre>USERWEIGHT 10</pre>

USESYSLOG	
<b>Format</b>	<BOOLEAN>[<FACILITY>]
<b>Default</b>	<i>FALSE:daemon</i>
<b>Description</b>	Specifies whether or not the scheduler will report key events to the system syslog facility. If the <FACILITY> is specified, Moab will report events to this syslog facility. See <a href="#">Logging Facilities</a> for more information.
<b>Example</b>	<pre>USESYSLOG TRUE:local3</pre> <i>Moab will report key events, commands, and failures to syslog using the local3 facility.</i>

USESYSTEMQUEUETIME	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether or not job prioritization should be based on the time the job has been eligible to run, i.e., idle and meets all fairness policies ( <i>TRUE</i> ) or the time the job has been idle ( <i>FALSE</i> ). See <a href="#">Priority Factors</a> for more info. <b>Note:</b> This parameter has been superseded by the <a href="#">JOBPRIOACCRUALPOLICY</a> parameter.
<b>Example</b>	<pre>USESYSTEMQUEUETIME FALSE</pre> <p><i>The queue time and expansion factor components of a job's priority will be calculated based on the length of time the job has been in the idle state.</i></p>

USEUSERHASH	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Enables searching of the user buffer using the user hash key instead of doing sequential searches of the user buffer.
<b>Example</b>	<pre>USEUSERHASH TRUE</pre>

VMCALCULATELOADBYVMSUM	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	When false, vmmigrate using overcommits uses the CPU load from the node to determine if VM's need to be migrated off the hypervisor. When true, overcommit vmmigrates calculates the total node load using the total sum reported by each VM on the hypervisor.
<b>Example</b>	<pre>VMCALCULATELOADBYVMSUM TRUE</pre>

VMCPURGETIME	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	5:00
<b>Description</b>	<p>When a VM completes, Moab stores it in a completed VM table for the specified amount of time. This prevents it from starting again if an RM reports it late. It also prevents a user from creating a VM with the same ID for a certain amount of time.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p><b>i</b> The VM will remain in the completed VM table for more than the specified amount of time if <b>VMSTALETIME</b> is greater than <b>VMCPURGETIME</b>. Both parameters must expire before Moab will remove the VM from the table.</p> </div>
<b>Example</b>	<div style="border: 1px dashed #000; border-radius: 10px; padding: 10px; margin-bottom: 10px;"> <pre>VMCPURGETIME 10:00</pre> </div> <p><i>Moab holds completed VMs for 10 minutes to prevent a late RM from reporting and restarting it.</i></p>

VMMIGRATETOZERoloadNODES	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Allows VM migrations to occur to and from hypervisors that do not report a CPUload or memory load.
<b>Example</b>	<div style="border: 1px dashed #000; border-radius: 10px; padding: 10px;"> <pre>VMMIGRATETOZERoloadNODES TRUE</pre> </div>

VMMIGRATETHROTTLE	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Sets the maximum allowable 'VM migrate' jobs at any given time.
<b>Example</b>	<pre>VMMIGRATETHROTTLE 20</pre> <p><i>Only 20 VM migrate jobs are allowed in the system at any given time.</i></p>

VMMIGRATIONPOLICY	
<b>Format</b>	<STRING>; values include <i>CONSOLIDATION</i> and <i>OVERCOMMIT</i>
<b>Default</b>	<i>NONE</i>
<b>Description</b>	<p>Choose only one of these values:</p> <ul style="list-style-type: none"> <li>• <i>CONSOLIDATION</i>- If the <i>CONSOLIDATION</i> flag is set, Moab consolidates VMs to allow nodes to go idle. This flag also ensures that no hypervisors are overloaded.</li> <li>• <i>OVERCOMMIT</i>- If the <i>OVERCOMMIT</i> flag is set, VMs to be migrated will be selected from overloaded hypervisors to bring them below the selected thresholds. This flag must be set for the <a href="#">VMOCTHRESHOLD</a> parameter to function.</li> </ul>
<b>Example</b>	<pre>VMMIGRATIONPOLICY OVERCOMMIT</pre>

VMMINOPDELAY	
<b>Format</b>	[HH[:MM[:SS]]
<b>Default</b>	---
<b>Description</b>	The minimum time between automatic VM node operations, such as creating, modifying, and destroying VMs. May prevent thrashing.
<b>Example</b>	<pre>VMMINOPDELAY 30</pre>

VMOCTHRESHOLD	
<b>Format</b>	<i>MEM:&lt;0-1&gt;,PROCS:&lt;0-1&gt;,DISK:&lt;0-1&gt;,SWAP:&lt;0-1&gt;,GMETRIC:&lt;metric&gt;:value</i>
<b>Default</b>	---
<b>Description</b>	Percentage threshold at which Moab begins to migrate virtual machines to other nodes. <a href="#">VMMIGRATIONPOLICY</a> must be set to OVERCOMMIT for this to occur.
<b>Example</b>	<pre> NODECFG[DEFAULT]  VMOCTHRESHOLD=PROC:.7,MEM:.9,GMETRIC:mem_io:6000  # This is the                     default global policy NODECFG[node42]   VMOCTHRESHOLD=PROC:.2,MEM:.1,GMETRIC:mem_io:12000 # This is a                     node-specific policy for node42                 </pre> <p><i>When a node surpasses .7 (70%) load of CPU or .9 (90%) of memory, Moab begins to migrate virtual machines to other nodes. When node42 surpasses .2 (20%) load of CPU or .1 (10%) of memory, Moab begins to migrate virtual machines to other nodes.</i></p>

VMPROVISIONSTATUSREADYVALUE	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Checks a VM for a special value or values (which Moab gets from the resource manager) and, based on the value, tells Moab that a VM was created..
<b>Example</b>	<pre> VMProvisionStatusReadyValue 2 VMProvisionStatusReadyValue 1-4,6,16                 </pre>

VMSARESTATIC	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<i>FALSE</i>
<b>Description</b>	When set to true, informs Moab that it can schedule under the assumption that no VMs will be migrated and no new VMs will be created, and disables Moab from scheduling any VM creations or migrations.
<b>Example</b>	<pre> VMSARESTATIC TRUE                 </pre>

VMSTALEACTION	
<b>Format</b>	One of the following: <i>IGNORE</i> , <i>CANCELTRACKINGJOB</i> , or <i>DESTROY</i>
<b>Default</b>	<i>IGNORE</i>
<b>Description</b>	<p>Specifies the action that is applied to a stale VM, or a VM that the resource manager has not reported to Moab recently (see <a href="#">VMSTALETIME</a>).</p> <ul style="list-style-type: none"> <li>• <i>IGNORE</i> (default) specifies that Moab will take no action.</li> <li>• <i>CANCELTRACKINGJOB</i> specifies that Moab will remove the tracking job for stale VMs, but will not remove the actual VM (not recommended).</li> <li>• <i>DESTROY</i> specifies that Moab destroys stale VMs.</li> </ul> <div style="border: 1px solid #0056b3; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p> If you specify <i>DESTROY</i>, you must also set the <a href="#">ENABLEVMDESTROY</a> parameter to TRUE.</p> </div>
<b>Example</b>	<code>VMSTALEACTION DESTROY</code>

VMSTALETIME	
<b>Format</b>	<code>[[HH:]MM:]SS</code>
<b>Default</b>	<i>10:00</i>
<b>Description</b>	<p>Specifies the amount of time a VM must be unreported by any resource manager before it is considered "stale."</p> <p>To specify what happens with the VM after it has become stale, see <a href="#">VMSTALEACTION</a>.</p>
<b>Example</b>	<code>VMSTALETIME 5:00</code> <div style="border: 1px dashed #ccc; padding: 5px; margin-top: 5px;"> <p><i>5 minutes must pass without a resource manager reporting a VM for it to be considered stale.</i></p> </div>

VMSTORAGEMOUNTDIR	
<b>Format</b>	<PATH>
<b>Default</b>	---
<b>Description</b>	The specified path is used as the default location for storage mounts in all newly created VMs (created via the <a href="#">mvmctl</a> command). This parameter defines the default storage mount directory if one is not specified.
<b>Example</b>	<pre>VMSTORAGEMOUNTDIR /var/spool</pre> <p><i>Moab uses <code>/var/spool</code> as a storage mount directory if a storage directory is not submitted (but additional storage is requested) at VM creation.</i></p>

VMTRACKING	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	When set to <b>TRUE</b> , VMTracking jobs are used to represent VMs in the queue.
<b>Example</b>	<pre>VMTRACKING TRUE</pre>

WALLTIMECAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum total pre-weighted absolute contribution to job priority which can be contributed by the walltime component. This value is specified as an absolute priority value, not as a percent.
<b>Example</b>	<pre>WALLTIMECAP 10000</pre> <p><i>Moab will bound a job's pre-weighted walltime priority component within the range +/- 10000.</i></p>

WALLTIMEWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight to be applied to the amount of walltime requested by a job (in seconds) (see <a href="#">Resource (RES) Factor</a> ).
<b>Example</b>	<pre>RESWEIGHT      10 WALLTIMEWEIGHT 100</pre> <p><i>Increase the priority of longer duration jobs.</i></p>

WCACCURACYCAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum total pre-weighted absolute contribution to job priority which can be contributed by the wallclock accuracy component. This value is specified as an absolute priority value, not as a percent.
<b>Example</b>	<pre>WCACCURACYCAP 10000</pre> <p><i>Moab will bound a job's pre-weighted wallclock accuracy priority component within the range +/- 10000.</i></p>

WCACCURACYWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority weight to be applied to the job's historical user wallclock accuracy (range 0.0 to 1.0) (see <a href="#">Fairshare (FS) Factor</a> ).
<b>Example</b>	<pre>FSWEIGHT      10 WCACCURACYWEIGHT 100</pre> <p><i>Favor jobs with good wallclock accuracies by giving them a priority increase.</i></p>

WCVIOLATIONACTION	
<b>Format</b>	one of <i>CANCEL</i> or <i>PREEMPT</i>
<b>Default</b>	<i>CANCEL</i>
<b>Description</b>	Specifies the action to take when a job exceeds its wallclock limit. If set to <i>CANCEL</i> , the job will be terminated. If set to <i>PREEMPT</i> , the action defined by <a href="#">PREEMPTPOLICY</a> parameter will be taken. See <a href="#">JOBMAXOVERRUN</a> or <a href="#">Usage-based limits</a> .
<b>Example</b>	<pre>WCVIOLATIONACTION  PREEMPT PREEMPTPOLICY      REQUEUE</pre> <p><i>Moab will requeue jobs which exceed their wallclock limit.</i></p>

WEBSERVICESURL	
<b>Format</b>	<URL>
<b>Default</b>	---
<b>Description</b>	If specified, Moab sends data to Moab Web Services (MWS) to be stored in a database. This allows Moab to spend more cycles on scheduling instead of database interaction. The sending occurs via HTTP PUT.
<b>Example</b>	<pre>WEBSERVICESURL http://mws-staging.ac:8080/mws/rm/moab/dump</pre> <p><i>Moab sends data that needs to be stored in a database to the specified URL.</i></p>

WIKIEVENTS	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	When set to true, Moab events are set to native wiki format (ATTR=VALUE pairs) to facilitate easier readability .
<b>Example</b>	<pre>WIKIEVENTS TRUE</pre> <p>Moab events will generate output in the format of the following sample:</p> <pre>09:26:40 1288279600:5 job 58 JOBEND 58 REQUESTEDNC=1 REQUESTEDTC=3 UNAME=wightman GNAME=wightman WCLIMIT=60 STATE=Completed RCLASS=[batch:1] SUBMITTIME=1288279493 RMEMCMP=&gt;= RDISKCMP=&gt;= RFEATURES=[NONE] SYSTEMQUEUEUETIME=1288279493 TASKS=1 FLAGS=RESTARTABLE PARTITION=pbs DPROCS=1 ENDDATE=2140000000 TASKMAP=proxy,GLOBAL SRM=pbs EXITCODE=0 SID=2357 NODEALLOCATIONPOLICY=SHARED EFFECTIVEQUEUEEDURATION=107</pre>

XFACTORCAP	
<b>Format</b>	<DOUBLE>
<b>Default</b>	0 (NO CAP)
<b>Description</b>	Specifies the maximum total pre-weighted absolute contribution to job priority which can be contributed by the expansion factor component. This value is specified as an absolute priority value, not as a percent.
<b>Example</b>	<pre>XFACTORCAP 10000</pre> <p>Moab will bound a job's pre-weighted XFactor priority component within the range +/- 10000.</p>

XFACTORWEIGHT	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the weight to be applied to a job's minimum expansion factor before it is added to the job's cumulative priority.
<b>Example</b>	<pre>XFACTORWEIGHT 1000</pre> <p><i>Moab will multiply a job's XFactor value by 1000 and then add this value to its total priority.</i></p>

XFMINWCLIMIT	
<b>Format</b>	[[[DD:]HH:]MM:]SS
<b>Default</b>	-1 (NO LIMIT)
<b>Description</b>	Specifies the minimum job wallclock limit that will be considered in job expansion factor priority calculations.
<b>Example</b>	<pre>XFMINWCLIMIT 0:01:00</pre> <p><i>Jobs requesting less than 1 minute of wallclock time will be treated as if their wallclock limit was set to 1 minute when determining expansion factor for priority calculations.</i></p>

## Appendix B: Multi-OS Provisioning

- [xCAT Configuration Requirements](#)
- [MSM Installation](#)
- [Integrating MSM and xCAT](#)
- [MSM Configuration](#)
- [Configuration Validation](#)
- [Troubleshooting](#)
- [Deploying Images with TORQUE](#)
- [Installing Moab on the Management Node](#)

- [Moab Configuration File Example](#)
- [Verifying the Installation](#)
- [xCAT Plug-in Configuration Parameters](#)

## Introduction

Moab can dynamically provision compute machines to requested operating systems and power off compute machines when not in use. Moab can intelligently control xCAT and use its advanced system configuration mechanisms to adapt systems to current workload requirements. Moab communicates with xCAT using the Moab Service Manager (MSM). MSM is a translation utility that resides between Moab and xCAT and acts as aggregator and interpreter. The Moab Workload Manager will query MSM, which in turn queries xCAT, about system resources, configurations, images, and metrics. After learning about these resources from MSM, Moab then makes intelligent decisions about the best way to maximize system utilization.

In this model Moab gathers system information from two resource managers. The first is TORQUE, which handles the workload on the system; the second is MSM, which relays information gathered by xCAT. By leveraging these software packages, Moab intelligently adapts clusters to deliver on-site goals.

This document assumes that xCAT has been installed and configured. It describes the process of getting MSM and xCAT communicating, and it offers troubleshooting guidance for basic integration. This document offers a description for how to get Moab communicating with MSM and the final steps in verifying a complete software stack.

## xCAT Configuration Requirements

Observe the following xCAT configuration requirements before installing MSM:

- Configure xCAT normally for your site.
  - Test the following commands to verify proper function:
    - **rpower**
    - **nodeset**
    - **makedhcp**
    - **makedns**
    - **nodestat**
    - **rvitals**
  - If MSM will run on a different machine than the one on which xCAT runs, install the xCAT client packages on that machine, and test the previously listed commands on that machine as well.
  - Configure and test all stateful/stateless images you intend to use.
- Configure xCAT to use either PostgreSQL or MySQL. Note that the default of SQLite may not function properly when MSM drives xCAT.

- PostgreSQL: See [xCATSetupPostgreSQL.pdf](#) for more information.
- MySQL: See [xCAT2.SetupMySQL.pdf](#) for more information.

**i** You must have a valid Moab license file (`moab.lic`) with provisioning and green enabled. For information on acquiring an evaluation license, please contact [info@adaptivecomputing.com](mailto:info@adaptivecomputing.com).

## MSM Installation

- Determine the installation directory (usually `/opt/moab/tools/msm`)
- Untar the MSM tarball into the specified directory (making it the MSM home directory, or `$MSMHOMEDIR`)
- Verify the required Perl modules and version are available

```
> perl -e 'use Storable 2.18'
> perl -MXML::Simple -e 'exit'
> perl -MProc::Daemon -e 'exit'
> perl -MDBD::SQLite -e 'exit'
```

## Integrating MSM and xCAT

Copy the `x_msm` table schema to the xCAT schema directory:

```
> cp $MSMHOMEDIR/contrib/xcat/MSM.pm $XCATROOT/lib/perl/xCAT_schema
```

Restart `xcatd` and check the `x_msm` table is correctly created:

```
> service xcatd restart
```

```
> tabdump x_msm
```

Prepare xCAT images and ensure they provision correctly (see xCAT documentation)

Populate the `x_msm` table with your image definitions:

```
> tabedit x_msm
#flavorname,arch,profile,os,nodeset,features,vmoslist,hvtype,hvgroupname,vmgroupname,comments,disable
"compute","x86_64","compute","centos5.3","netboot","torque",,,,,,
"science","x86","compute","scientific_linux","netboot","torque",,,,,,
```

- **flavorname** - A user specified name for the image and settings; also an xCAT group name, nodes are added to this group when provisioned
- **arch** - Architecture as used by xCAT
- **profile** - Profile as used by xCAT
- **os** - Operating system as used by xCAT
- **nodeset** - One of netboot|install|stelite

- **features** - Names of xCAT groups that identify special hardware features ('torque' and 'paravirt' are special cases)
- **vmoslist** - Note: Not used. List of flavorname's this image may host as VMs (hypervisor images only)
- **hvtype** - Note: Not used. One of esx|xen|kvm (hypervisor images only)
- **hvgroupname** - Note: Not used. Name of xCAT group nodes will be added to when provisioned to this image
- **vmgroupname** - Note: Not used. Name of xCAT group VMs will be added to when hosted on a hypervisor of this image
- **comments** - User specified comments
- **disable** - Flag to temporarily disable use of this image

Ensure all xCAT group names in the `x_msm` table exist in the xCAT nodegroup table

```
> tabedit nodegroup
```

Edit as necessary to simulate the following example:

```
#groupname, grouptype, members, wherevals, comments, disable
"compute",,,,,,
"esxi4",,,,,,
"esxhv",,,,,,
"esxvmngt",,,,,,
```

After making any necessary edits, run the following command:

```
> nodels compute,esxi4,esxhv,esxvmngt
# should complete without error, ok if doesn't return anything
```

## MSM Configuration

Edit `$MSMHOMEDIR/msm.cfg` and configure the xCAT plug-in. Below is a generic example for use with TORQUE without virtualization. See the section on configuration parameters for a complete list of parameters and descriptions.

```
# MSM configuration options
RMCFG[msm]      PORT=24603
RMCFG[msm]      POLLINTERVAL=45
RMCFG[msm]      LOGFILE=/opt/moab/log/msm.log
RMCFG[msm]      LOGLEVEL=8
RMCFG[msm]      DEFAULTNODEAPP=xcat

# xCAT plugin specific options
APPCFG[xcat]    DESCRIPTION="xCAT plugin"
APPCFG[xcat]    MODULE=Moab::MSM::App::xCAT
APPCFG[xcat]    LOGLEVEL=3
APPCFG[xcat]    POLLINTERVAL=45
APPCFG[xcat]    TIMEOUT=3600
APPCFG[xcat]    _USEOPIDS=0
APPCFG[xcat]    _NODERANGE=moab,esxcompute
APPCFG[xcat]    _USESTATES=boot,netboot,install
APPCFG[xcat]    _LIMITCLUSTERQUERY=1
APPCFG[xcat]    _RPOWERTIMEOUT=120
APPCFG[xcat]    _DONODESTAT=1
APPCFG[xcat]    _REPORTNETADDR=1
APPCFG[xcat]    _CQXCATSESSIONS=4
```

## Configuration Validation

Set up environment to manually call MSM commands:

```
# substitute appropriate value(s) for path(s)
export MSMHOMEDIR=/opt/moab/tools/msm
export MSMLIBDIR=/opt/moab/tools/msm
export PATH=$PATH:/$MSMLIBDIR/contrib:$MSMLIBDIR/bin
```

Verify that MSM starts without errors:

```
> msmd
```

Verify that the expected nodes are listed, without errors, using the value of **\_NODERANGE** from `msm.cfg`.

```
> nodels <_NODERANGE>
```

Verify that the expected nodes, are listed in the cluster query output from MSM:

```
> cluster.query.pl
```

Provision all nodes through MSM for the first time (pick and image name from `x_msm`):

```
> for i in `nodels <_NODERANGE>`; do node.modify.pl $i --set os=<image_name>;done
```

Verify the nodes correctly provision and that the correct OS is reported (which may take some time after the provisioning requests are made):

```
> cluster.query.pl
```

## Troubleshooting

- **msmctl -a does not report the xCAT plugin** - Check the log file (path specified in `msm.cfg`) for error messages. A common cause is missing Perl modules (Storable, DBD::SQLite, xCAT::Client).
- **cluster.query.pl does not report any nodes** - Check that the xCAT command `nodels <noderange>`, where `<noderange>` is the value configured for `_NODERANGE` in `msm.cfg`, outputs the nodes expected.
- **cluster.query.pl does not report OS** - MSM must provision a node to recognize what the current operating system is. It is not sufficient to look up the values in the `nodetype` table because MSM has no way of recognizing whether `nodeset` and `rpower` were run with the current values in the `nodetype` table.
- **cluster.query.pl does not report OSLIST, or does not report the expected OSLIST for a node** - Check that the node belongs to the appropriate groups, particularly any listed in the `features` field of the `x_msm` table for the missing image name.

## Deploying Images with TORQUE

When using MSM + xCAT to deploy images with TORQUE, there are some special configuration considerations. Most of these also apply to other workload resource managers.

Note that while the MSM xCAT plugin contains support for manipulating TORQUE directly, this is not an ideal solution. If you are using a version of xCAT that supports prescripts, it is more appropriate to write prescripts that manipulate TORQUE based on the state of the xCAT tables. This approach is also applicable to other workload resource managers, while the xCAT plugin only deals with TORQUE.

Several use cases and configuration choices are discussed in what follows.

Each image should be configured to report its image name through TORQUE. In the TORQUE `pbs_mom_mom_config` file the `opsys` value should mirror the name of the image. See [Appendix C: Node manager \(MOM\) configuration on page 2783](#) in the TORQUE Administrator's Guide for more information.

## Installing Moab on the Management Node

Moab is the intelligence engine that coordinates the capabilities of xCAT and TORQUE to dynamically provision compute nodes to the requested operating system. Moab also schedules workload on the system and powers off idle nodes. [Download](#) and [install](#) Moab.

## Moab Configuration File Example

Moab stores its configuration in the `moab.cfg` file: `/opt/moab/etc/moab.cfg`. A sample configuration file, set up and optimized for adaptive computing follows:

```

SCHEDCFG[Moab]          SERVER=gpc-sched:42559
ADMINCFG[1]             USERS=root,egan
LOGLEVEL                7

# How often (in seconds) to refresh information from TORQUE and MSM
RMPOLLINTERVAL         60,60
RESERVATIONDEPTH       10
DEFERTIME               0
TOOLS DIR              /opt/moab/tools

#####
# TORQUE and MSM configuration                                     #
#####
RMCFG[torque]          TYPE=PBS
RMCFG[msm]             TYPE=NATIVE:msm FLAGS=autosync,NOCREATERESOURCE RESOURCETYPE=PROV
RMCFG[msm]             TIMEOUT=60
RMCFG[msm]             PROVDURATION=10:00
AGGREGATENODEACTIONS   TRUE

#####
# ON DEMAND PROVISIONING SETUP                                   #
#####
QOSCFG[od]             QFLAGS=PROVISION
USERCFG[DEFAULT]       QLIST=od
NODEALLOCATIONPOLICY    PRIORITY
NODECFG[DEFAULT]       PRIORITYF=1000*OS+1000*POWER
NODEAVAILABILITYPOLICY DEDICATED
CLASSCFG[DEFAULT]      DEFAULT.OS=scinetcompute

#####
# GREEN POLICIES   #
#####
NODECFG[DEFAULT]       POWERPOLICY=ONDEMAND
PARCFG[ALL]            NODEPOWEROFFDURATION=20:00
NODEIDLEPOWERTHRESHOLD 600
# END Example moab.cfg

```

## Verifying the Installation

When Moab starts it immediately communicates with its configured resource managers. In this case Moab communicates with TORQUE to get compute node and job queue information. It then communicates with MSM to determine the state of the nodes according to xCAT. It aggregates this information and processes the jobs discovered from TORQUE.

When a job is submitted, Moab determines whether nodes need to be provisioned to a particular operating system to satisfy the requirements of the job. If any nodes need to be provisioned Moab performs this action by creating a provisioning system job (a job that is internal to Moab). This system job communicates with xCAT to provision the nodes and remain active while the nodes are provisioning. Once the system job has provisioned the nodes it informs the user's job that the nodes are ready at which time the user's job starts running on the newly provisioned nodes.

When a node has been idle for a specified amount of time (see [NODEIDLEPOWERTHRESHOLD](#)), Moab creates a power-off system job. This job communicates with xCAT to power off the nodes and remains active in the job queue until the nodes have powered off. Then the system job informs Moab that the nodes are powered off but are still available to run jobs. The power off system job then exits.

To verify correct communication between Moab and MSM run the `mdiag -R -v msm` command.

```

$ mdiag -R -v msm
diagnosing resource managers
RM[msm]      State: Active  Type: NATIVE:MSM  ResourceType: PROV
Timeout:      30000.00 ms
Cluster Query URL: $HOME/tools/msm/contrib/cluster.query.xcat.pl
Workload Query URL: exec://$TOOLSDIR/msm/contrib/workload.query.pl
Job Start URL:  exec://$TOOLSDIR/msm/contrib/job.start.pl
Job Cancel URL:  exec://$TOOLSDIR/msm/contrib/job.modify.pl
Job Migrate URL: exec://$TOOLSDIR/msm/contrib/job.migrate.pl
Job Submit URL:  exec://$TOOLSDIR/msm/contrib/job.submit.pl
Node Modify URL: exec://$TOOLSDIR/msm/contrib/node.modify.pl
Node Power URL:  exec://$TOOLSDIR/msm/contrib/node.power.pl
RM Start URL:   exec://$TOOLSDIR/msm/bin/msmd
RM Stop URL:    exec://$TOOLSDIR/msm/bin/msmctl?-k
System Modify URL: exec://$TOOLSDIR/msm/contrib/node.modify.pl
Environment:
MSMHOMEDIR=/home/wightman/test/scinet/tools//msm;MSMLIBDIR=/home/wightman/test/scinet/
tools//msm
Objects Reported:  Nodes=10 (0 procs)  Jobs=0
Flags:             autosync
Partition:         SHARED
Event Management:  (event interface disabled)
RM Performance:    AvgTime=0.10s  MaxTime=0.25s  (38 samples)
RM Languages:      NATIVE
RM Sub-Languages:  -

```

To verify nodes are configured to provision use the `checknode -v` command. Each node will have a list of available operating systems.

```

$ checknode n01
node n01
State:      Idle (in current state for 00:00:00)
Configured Resources: PROCS: 4  MEM: 1024G  SWAP: 4096M  DISK: 1024G
Utilized Resources: ---
Dedicated Resources: ---
Generic Metrics:    watts=25.00,temp=40.00
Power Policy:       Green (global policy)  Selected Power State: Off
Power State:        Off
Power:              Off
MTBF(longterm):    INFINITY  MTBF(24h):  INFINITY
Opsys:              compute  Arch:      ---
OS Option:          compute
OS Option:          computea
OS Option:          gpfscmpute
OS Option:          gpfscmputea
Speed:              1.00      CPULoad:    0.000
Flags:              rmdetected
RM[msm]:            TYPE=NATIVE:MSM  ATTRO=POWER
EffNodeAccessPolicy: SINGLEJOB
Total Time: 00:02:30  Up: 00:02:19 (92.67%)  Active: 00:00:11 (7.33%)

```

To verify nodes are configured for Green power management, run the `mdiag -G` command. Each node will show its power state.

```

$ mdiag -G
NOTE: power management enabled for all nodes
Partition ALL: power management enabled
Partition NodeList:
Partition local: power management enabled
Partition NodeList:
node n01 is in state Idle, power state On (green powerpolicy enabled)
node n02 is in state Idle, power state On (green powerpolicy enabled)
node n03 is in state Idle, power state On (green powerpolicy enabled)
node n04 is in state Idle, power state On (green powerpolicy enabled)
node n05 is in state Idle, power state On (green powerpolicy enabled)
node n06 is in state Idle, power state On (green powerpolicy enabled)
node n07 is in state Idle, power state On (green powerpolicy enabled)
node n08 is in state Idle, power state On (green powerpolicy enabled)
node n09 is in state Idle, power state On (green powerpolicy enabled)
node n10 is in state Idle, power state On (green powerpolicy enabled)
Partition SHARED: power management enabled

```

To submit a job that dynamically provisions compute nodes, run the `msub -l os=<image>` command.

```

$ msub -l os=computea job.sh
yuby.3
$ showq
active jobs-----
JOBID            USERNAME        STATE PROCS    REMAINING      STARTTIME
provision-4      root            Running      8      00:01:00  Fri Jun 19 09:12:56
1 active job
                  8 of 40 processors in use by local jobs (20.00%)
                  2 of 10 nodes active          (20.00%)

eligible jobs-----
JOBID            USERNAME        STATE PROCS    WCLIMIT        QUEUETIME
yuby.3           wightman        Idle         8      00:10:00  Fri Jun 19 09:12:55
1 eligible job

blocked jobs-----
JOBID            USERNAME        STATE PROCS    WCLIMIT        QUEUETIME

0 blocked jobs
Total jobs: 2

```

Notice that Moab created a provisioning system job named `provision-4` to provision the nodes. When `provision-4` detects that the nodes are correctly provisioned to the requested OS, the submitted job `yuby.3` runs:

```

$ showq
active jobs-----
JOBID            USERNAME        STATE PROCS    REMAINING      STARTTIME
yuby.3           wightman        Running      8      00:08:49  Fri Jun 19 09:13:29
1 active job
                  8 of 40 processors in use by local jobs (20.00%)
                  2 of 10 nodes active          (20.00%)

eligible jobs-----
JOBID            USERNAME        STATE PROCS    WCLIMIT        QUEUETIME

0 eligible jobs

blocked jobs-----
JOBID            USERNAME        STATE PROCS    WCLIMIT        QUEUETIME

0 blocked jobs
Total job: 1

```

The `checkjob` command shows information about the provisioning job as well as the submitted job. If any errors occur, run the `checkjob -v <jobid>` command to diagnose failures.

## xCAT Plug-in Configuration Parameters

Plugin parameters that begin with an underscore character are specific to the xCAT plug-in; others are common to all plug-ins and may either be set in the **RMCFG[msm]** for all plug-ins, or per plug-in in the **APPCFG[<plugin\_name>]**.

<u>Description</u>	<u>FeatureGroups</u>	<u>VerifyRPower</u>
<u>Module</u>	<u>DefaultVMCProc</u>	<u>RPowerTimeOut</u>
<u>LogLevel</u>	<u>DefaultVMDisk</u>	<u>QueueRPower</u>
<u>PollInterval</u>	<u>DefaultVMCMemory</u>	<u>RPowerQueueAge</u>
<u>TimeOut</u>	<u>KVMStoragePath</u>	<u>RPowerQueueSize</u>
<u>NodeRange</u>	<u>ESXStore</u>	<u>MaskOSWhenOff</u>
<u>CQxCATSessions</u>	<u>ESXCFGPath</u>	<u>ModifyTORQUE</u>
<u>DORVitals</u>	<u>VMInterfaces</u>	<u>ReportNETADDR</u>
<u>PowerString</u>	<u>XenHostInterfaces</u>	<u>UseOpIDs</u>
<u>DoNodeStat</u>	<u>KVMHostInterfaces</u>	<u>VMIPRange</u>
<u>DoxCATStats</u>	<u>VMSovereign</u>	<u>xCATHost</u>
<u>LockDir</u>	<u>UseStates</u>	<u>NoRollbackOnError</u>
<u>HVxCATPasswdKey</u>	<u>ImagesTabName</u>	

Description	
<b>Format</b>	Double quoted string containing brief description of plugin.
<b>Default</b>	---
<b>Description</b>	This information is not visible in Moab, but shows up in <code>msmctl -a</code> .

Module	
<b>Format</b>	Moab::MSM::App::xCAT
<b>Default</b>	---
<b>Description</b>	Name of the plugin module to load.

LogLevel	
<b>Format</b>	1-9
<b>Default</b>	5

LogLevel	
<b>Description</b>	Used to control the verbosity of logging, 1 being the lowest (least information logged) and 9 being the highest (most information logged). For initial setup and testing, 8 is recommended, then lowering to 3 (only errors logged) for normal operation. Use 9 for debugging, or when submitting a log file for support.

PollInterval	
<b>Format</b>	Integer > 0
<b>Default</b>	60
<b>Description</b>	MSM will query xCAT every <b>POLLINTERVAL</b> seconds to update general node status. This number will likely require tuning for each specific system. In general, to develop this number, you should pick a fraction of the total nodes MSM will be managing ( $1/_CQXCATSESSIONS$ ), and time how long it takes run nodestat, rpower stat, and optionally rvitals on these nodes, and add ~15%. Increasing the <b>POLLINTERVAL</b> will lower the overall load on the xCAT headnode, but decrease the responsiveness to provisioning and power operations.

TimeOut	
<b>Format</b>	Integer value > POLLINTERVAL
<b>Default</b>	300
<b>Description</b>	This parameter controls how long MSM will wait for child processed to complete (all xCAT commands are run in child processes). After <b>TIMEOUT</b> seconds, if a child has not returned it will be killed, and an error reported for the operation.

_NodeRange	
<b>Format</b>	Any valid noderange (see the xCAT noderange man page).
<b>Default</b>	All
<b>Description</b>	When MSM queries xCAT this is the noderange it will use. At sites where xCAT manages other hardware that Moab is not intended to control, it is important to change this.

<b>_CQxCATSessions</b>	
<b>Format</b>	Positive integer > 1
<b>Default</b>	<i>10</i>
<b>Description</b>	MSM will divide the node list generated by <code>nodels</code> into this many groups and simultaneously query xCAT for each group. The value may need tuning for large installations, higher values will cause the time to complete a single cluster query to go down, but cause a higher load on the xCAT headnode.

<b>_DORVitals</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>0</i>
<b>Description</b>	When set to <i>1</i> , MSM will poll rvitals power and led status (see the xCAT rvitals man page). This only works with IBM BMCs currently. In order to use this, xCAT should respond without error to the <code>rvitals &lt;noderange&gt; watts</code> and <code>rvitals &lt;noderange&gt; leds</code> commands. Status is reported as <b>GMETR TIC[watts]</b> and <b>GMETR IC[leds]</b> . See also the <a href="#">_PowerString on page 1090</a> configuration parameter.

<b>_PowerString</b>	
<b>Format</b>	single quote delimited string
<b>Default</b>	<i>'AC Avg Power'</i>
<b>Description</b>	Only meaningful when used with <a href="#">_DORVitals on page 1090</a> =1. Some BMCs return multiple responses to the rvitals command, or use slightly different text to describe the power metrics. Use this parameter to control what is reported to Moab. You can use <code>'\$MSMLIBDIR/contrib/xcat/dump.xcat.cmd.pl rvitals &lt;node_name&gt; power'</code> and examine the output to determine what the appropriate value of this string is.

<b>_DoNodeStat</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>1</i>

**\_DoNodeStat**

<b>Description</b>	If set to <i>0</i> , MSM will not call <code>nodestat</code> to generate a substate. This can be used to speed up the time it takes to query xCAT, and you do not need the substate visible to Moab.
--------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_DoxCATStats**

<b>Format</b>	<i>0</i> or <i>1</i>
---------------	----------------------

<b>Default</b>	<i>0</i>
----------------	----------

<b>Description</b>	If Set to <i>1</i> , MSM will track performance statistics about calls to xCAT, and the performance of higher level operations. The information is available via the script <code>\$MSMHOMEDIR/contrib/xcat/xcatstats.pl</code> . This parameter is useful for tuning the <b>POLLINTERVAL</b> and <b>CQxCATSessions</b> on page 1090 configuration parameters.
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_LockDir**

<b>Format</b>	Existing path on MSM host
---------------	---------------------------

<b>Default</b>	<i>\$MSMHOMEDIR/lck</i>
----------------	-------------------------

<b>Description</b>	This is a path to where MSM maintains lock files to control concurrency with some Xen and KVM operations.
--------------------	-----------------------------------------------------------------------------------------------------------

**\_HVxCATPasswdKey**

<b>Format</b>	key value in the xCAT passwd table
---------------	------------------------------------

<b>Default</b>	<i>vmware</i>
----------------	---------------

<b>Description</b>	This is where MSM gets the user/password to communicate with ESX hypervisors.
--------------------	-------------------------------------------------------------------------------

**\_FeatureGroups**

<b>Format</b>	Comma delimited string of xCAT group names.
---------------	---------------------------------------------

<b>Default</b>	<i>---</i>
----------------	------------

<b>_FeatureGroups</b>	
<b>Description</b>	MSM builds the OSLIST for a node as the intersection of <b>_FEATUREGROUPS</b> , features specified in <code>x_msm</code> for that image, and the nodes group membership. The value 'torque' is special, and indicates that the image uses TORQUE, and the node should be added/removed from TORQUE during provisioning when used in conjunction with the <a href="#">_ModifyTORQUE on page 1096</a> parameter.

<b>_DefaultVMCProc</b>	
<b>Format</b>	1-?
<b>Default</b>	1
<b>Description</b>	If not explicitly specified in the create request, MSM will create VMs with this many processors.

<b>_DefaultVMDisk</b>	
<b>Format</b>	Positive integer values, minimum is determined by your vm image needs
<b>Default</b>	4096
<b>Description</b>	If not explicitly specified in the create request, MSM will create VMs with this much disk allocated.

<b>_DefaultVMCMemory</b>	
<b>Format</b>	Positive integer values, minimum is determined by your vm image needs
<b>Default</b>	512
<b>Description</b>	If not specified, MSM will create VMs with this much memory allocated.

<b>_KVMStoragePath</b>	
<b>Format</b>	Existing path on MSM host
<b>Default</b>	<code>/vms</code>
<b>Description</b>	File backed disk location for stateful KVM VMS will be placed here.

<b>_ESXStore</b>	
<b>Format</b>	Mountable NFS Path
<b>Default</b>	---
<b>Description</b>	Location of ESX stores.

<b>_ESXCFGPath</b>	
<b>Format</b>	Mountable NFS Path
<b>Default</b>	<i>ESXStore</i>
<b>Description</b>	Location of ESX VM configuration files.

<b>_VMInterfaces</b>	
<b>Format</b>	Name of bridge device in your VM image
<b>Default</b>	<i>br0</i>
<b>Description</b>	Bridge device name passed to libvirt for network configuration of VMs (overrides _XENHOSTINTERFACES and _KVMHOSTINTERFACES if specified).

<b>_XenHostInterfaces</b>	
<b>Format</b>	Name of bridge device in your VM image
<b>Default</b>	<i>xenbr0</i>
<b>Description</b>	Bridge device name passed to libvirt for network configuration of Xen VMs.

<b>_KVMHostInterfaces</b>	
<b>Format</b>	Name of bridge device in your VM image
<b>Default</b>	<i>br0</i>

**\_KVMHostInterfaces**

<b>Description</b>	Bridge device name passed to libvirt for network configuration of KVM VMs.
--------------------	----------------------------------------------------------------------------

**\_VMsovereign**

<b>Format</b>	<i>0 or 1</i>
---------------	---------------

<b>Default</b>	<i>0</i>
----------------	----------

<b>Description</b>	Setting this attribute will cause Moab to reserve VMs' memory and procs on the hypervisor and treat the VM as the workload — additional workload cannot be scheduled on the VM.
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_UseStates**

<b>Format</b>	Valid xCAT chain.currstate values (see the xCAT chain man page)
---------------	-----------------------------------------------------------------

<b>Default</b>	<i>boot,netboot,install</i>
----------------	-----------------------------

<b>Description</b>	Nodes that do not have one of these values in the xCAT <code>chain.currstate</code> field will reported with <b>STATE=Updating</b> . Use this configuration parameter to prevent Moab from scheduling nodes that are updating firmware, etc.
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_ImagesTabName**

<b>Format</b>	Existing xCAT table that contains your image definitions.
---------------	-----------------------------------------------------------

<b>Default</b>	<i>x_msm</i>
----------------	--------------

<b>Description</b>	This table specifies the images that may be presented to Moab in a node's OSLIST. The xCAT schema for this table is defined in <code>\$MSMHOMEDIR/contrib/xcat/MSM.pm</code> , which needs to be copied to the <code>\$XCATROOT/lib/perl/xCAT_schema</code> directory.
--------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_VerifyRPower**

<b>Format</b>	<i>0 or 1</i>
---------------	---------------

<b>Default</b>	<i>0</i>
----------------	----------

**\_VerifyRPower**

<b>Description</b>	If set, MSM will attempt to confirm that rpower requests were successful by polling the power state with rpower stat until the node reports the expected state, or <a href="#">_RPowerTimeOut on page 1095</a> is reached. <b>NOTE:</b> This can create significant load on the xCAT headnode.
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**\_RPowerTimeOut**

<b>Format</b>	Positive integer values
<b>Default</b>	60
<b>Description</b>	Only meaningful when used with <a href="#">_VerifyRPower on page 1094</a> . If nodes do not report the expected power state in this amount of time, a GEVENT will be produced on the node (or system job).

**\_QueueRPower**

<b>Format</b>	0 or 1
<b>Default</b>	0
<b>Description</b>	When set, this parameter will cause MSM to aggregate rpower requests to xCAT into batches. The timing and size of these batches is controlled with the <a href="#">_RPowerQueueAge on page 1095</a> and <a href="#">_RPowerQueueSize on page 1096</a> parameters. <b>NOTE:</b> This can significantly reduce load on the xCAT headnode, but will cause the power commands to take longer, and MSM shutdown to take longer.

**\_RPowerQueueAge**

<b>Format</b>	Positive integer values
<b>Default</b>	30
<b>Description</b>	Only meaningful when used with <a href="#">_QueueRPower on page 1095</a> . MSM will send any pending rpower requests when the oldest request in the queue exceeds this value (seconds).

<b>_RPowerQueueSize</b>	
<b>Format</b>	Positive integer values
<b>Default</b>	<i>200</i>
<b>Description</b>	Only meaningful when used with <a href="#">_QueueRPower on page 1095</a> . MSM will send any pending rpower requests when the queue depth exceeds this value.

<b>_MaskOSWhenOff</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>0</i>
<b>Description</b>	When set, this parameter will cause MSM to report <i>OS=None</i> for nodes that are powered off. This may be useful when mixing stateless and stateful images, forcing Moab to request provisioning instead of just powering on a node.

<b>_ModifyTORQUE</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>0</i>
<b>Description</b>	When set, this parameter will cause MSM to add and removes nodes and VMs from TORQUE as required by provisioning. See the <a href="#">_FeatureGroups on page 1091</a> parameter as well.

<b>_ReportNETADDR</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>0</i>
<b>Description</b>	When set, this parameter will cause MSM to report NETADDR=<hosts.ip from xCAT>.

<b>_UseOpIDs</b>	
<b>Format</b>	<i>0 or 1</i>

<b>_UseOpIDs</b>	
<b>Default</b>	<i>0</i>
<b>Description</b>	When set, this parameter will cause errors to be reported as GEVENTs on the provided system job, instead of a node (Moab 5.4 only, with appropriate Moab CFG)

<b>_VMIPRange</b>	
<b>Format</b>	Comma separated list of dynamic ranges for VM (ex '10.10.23.100-200,10.10.24.1-255')
<b>Default</b>	---
<b>Description</b>	Use this parameter to specify a pool of IPs that MSM should assign to VMs at creation time. IPs are selected sequentially from this list as available. Omit this configuration parameter if an external service is managing IP assignment, or if they are all previously statically assigned.

<b>_xCATHost</b>	
<b>Format</b>	<code>&lt;xcat_headnode&gt;:&lt;xcatd_port&gt;</code>
<b>Default</b>	<i>localhost:3001</i>
<b>Description</b>	Use to configure MSM to communicate with xCAT on another host.

<b>_NoRollbackOnError</b>	
<b>Format</b>	<i>0 or 1</i>
<b>Default</b>	<i>0</i>
<b>Description</b>	When an error occurs and rollback is activated (as it is by default), rollback causes a reversion to the previous successful request. <b>_NoRollbackOnError</b> is useful for debugging to determine the xCAT state if no rollback occurred. If set to 1 and an error occurs between MSM and xCAT when creating a node, assigning a name (DNS) to a node, or assigning an IP address (DHCP) to a node, then no rollback occurs.

## Event Dictionary



See "[Logging Overview](#)" for more information about Moab logging.

### Moab Event Dictionary

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-0005</b>	USER	system.-moab	INFO	MWM_TESTING_INFO	Testing with argument1: %s. and argument2: %s and argument3: %s and argument4: %s	Internal error for testing diagnostics.
<b>0x100-0065</b>	USER	domain.lifecycle	INFO	MWM_JOB_CANCEL	Job %s was canceled. %s	The job was canceled.
<b>0x100-0066</b>	USER	domain.lifecycle	INFO	MWM_JOB_END_SUCCESSFUL	Job %s finished successfully. %s	The job finished successfully.
<b>0x100-0068</b>	USER	domain.lifecycle	INFO	MWM_JOB_USER_HOLD	Job %s had a user hold applied.	A user hold was applied to the job.
<b>0x100-0069</b>	USER	domain.lifecycle	INFO	MWM_JOB_SYSTEM_HOLD	Job %s had a system hold applied.	A system hold was applied to the job.
<b>0x100-006a</b>	USER	domain.lifecycle	INFO	MWM_JOB_BATCH_HOLD	Job %s had a batch hold applied.	A batch hold was applied to the job.
<b>0x100-006b</b>	USER	domain.lifecycle	INFO	MWM_JOB_DEFER_HOLD	Job %s had a defer hold applied.	A defer hold was applied to the job.
<b>0x100-006c</b>	USER	domain.lifecycle	INFO	MWM_JOB_MODIFY	Job %s was modified. %s	One of the attributes of the job was modified either via a user initiated action or an automated action.
<b>0x100-006d</b>	USER	domain.lifecycle	INFO	MWM_JOB_REJECT	Job %s was rejected. %s	The job was rejected for some reason.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-006e</b>	USER	domain.lifecycle	INFO	MWM_JOB_RELEASE	Job %s was released.	Any holds placed on the job have been released, and the job is not prevented from running due to any hold action. The job may still not be able to run due to other considerations.
<b>0x100-006f</b>	USER	domain.lifecycle	INFO	MWM_JOB_START	Job %s was started. %s	The job was started on its designated node[s].
<b>0x100-0070</b>	USER	domain.lifecycle	INFO	MWM_JOB_SUBMIT	Job %s was submitted. %s	The job has been submitted to Moab and is being evaluated and processed.
<b>0x100-0071</b>	USER	domain.lifecycle	INFO	MWM_JOB_CREATED	Job %s was created.	The job has been created and will be queued for execution.
<b>0x100-0072</b>	USER	domain.lifecycle	INFO	MWM_JOB_QUEUE	Job %s was requeued. %s	The job has been requeued so it can be executed again.
<b>0x100-0073</b>	USER	domain.lifecycle	INFO	MWM_JOB_CANCEL_CLEANUP_STARTED	Job %s is being cleaned up due to cancel request.	The job has been issued a cancel request and is being cleaned up.
<b>0x100-0074</b>	USER	domain.lifecycle	INFO	MWM_JOB_CLEANUP_STARTED	Job %s is being cleaned up.	The job has ended and is being cleaned up.
<b>0x100-0075</b>	USER	domain.lifecycle	INFO	MWM_JOB_DEFERRED	Job %s has been deferred.	The job has been deferred.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-00c9</b>	USER	domain.lifecycle	INFO	MWM_NODE_EVAC_VMS	Evacuating VMs off node %s.	Evacuating VMs off the node.
<b>0x100-012c</b>	USER	domain.lifecycle	INFO	MWM_RSV_CREATE	Reservation %s was created. %s	The reservation has been created and is stored in the system.
<b>0x100-012d</b>	USER	domain.lifecycle	INFO	MWM_RSV_START	Reservation %s has started.	The reservation has started.
<b>0x100-012e</b>	USER	domain.lifecycle	INFO	MWM_RSV_END	Reservation %s has ended.	The reservation has ended.
<b>0x100-0190</b>	USER	system-moab	INFO	MWM_SCHED_COMMAND	The following scheduler command was submitted: %s	External commands are submitted to Moab in a variety of ways. This event documents the command line and possibly other information associated with the command. These commands typically have the ability to change behavior/state within Moab. Commands that are typically queries are not included.
<b>0x100-0192</b>	USER	system-moab	INFO	MWM_SCHED_CYCLE_START	A scheduler iteration is beginning. %s	Moab periodically checks through submitted jobs and makes decisions regarding which jobs are scheduled. One of these iterations is beginning now.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-0193</b>	USER	system-moab	INFO	MWM_SCHED_CYCLE_END	A scheduler iteration is ending.	Moab periodically checks through submitted jobs and makes decisions regarding which jobs are scheduled. One of these iterations is ending now.
<b>0x100-0194</b>	USER	system-moab	INFO	MWM_SCHED_PAUSE	The scheduler has been paused. %s	The Moab scheduler has been administratively paused. New jobs may be submitted and existing jobs will continue to run, but no new jobs will be scheduled as long as Moab is paused.
<b>0x100-0195</b>	USER	system-moab	INFO	MWM_SCHED_RECYCLE	The scheduler has been recycled. %s	The Moab scheduler has been administratively recycled. The process will cleanly exit and save the state data. It will then restart, read in the data, and resume scheduling.
<b>0x100-0196</b>	USER	system-moab	INFO	MWM_SCHED_RESUME	The scheduler has been resumed.	The Moab scheduler has been administratively resumed. A new scheduling iteration will begin immediately and continue regularly.
<b>0x100-0197</b>	USER	system-moab	INFO	MWM_SCHED_START	The scheduler has started.	The Moab scheduler has started.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-0198</b>	USER	system-moab	INFO	MWM_SCHED_STOP	The scheduler has stopped. %s	The Moab scheduler has stopped.
<b>0x100-01f4</b>	USER	domain.lifecycle	INFO	MWM_TRIG_CREATE	Trigger %s has been created.	The named trigger has been created and is now recognized in the Moab system.
<b>0x100-01f5</b>	USER	domain.lifecycle	INFO	MWM_TRIG_START	Trigger %s has started.	The named trigger has started its action.
<b>0x100-01f6</b>	USER	domain.lifecycle	INFO	MWM_TRIG_END	Trigger %s has ended. %s	The named trigger has finished its action.
<b>0x100-01f8</b>	USER	domain.lifecycle	INFO	MWM_TRIG_THRESHOLD	Trigger %s threshold event: %s	A trigger threshold has been encountered. Additional details regarding the threshold may be included in the text.
<b>0x100-0258</b>	USER	domain.lifecycle	INFO	MWM_VM_SUBMIT	VM %s has been submitted.	The named VM has been submitted and is now recognized in the Moab system.
<b>0x100-0259</b>	USER	domain.lifecycle	INFO	MWM_VM_DESTROY	VM %s has been terminated.	The named VM has finished its lifecycle and is now removed and added to the completed table.
<b>0x100-025a</b>	USER	domain.lifecycle	INFO	MWM_VM_CANCEL	VM %s has been canceled.	The named VM has been canceled.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-025b</b>	USER	domain.lifecycle	INFO	MWM_VM_END	VM %s has been ended.	The named VM has been canceled because it has exceeded its allocated walltime.
<b>0x100-025c</b>	USER	domain.lifecycle	INFO	MWM_VM_MIGRATE_START	VM %s migration has started. (%s)	The named VM has started its migration. Additional information may be provided regarding source and destination nodes.
<b>0x100-025d</b>	USER	domain.lifecycle	INFO	MWM_VM_MIGRATE_END	VM %s migration has finished. (%s)	The named VM has finished its migration. Additional information may be provided regarding source and destination nodes.
<b>0x100-025f</b>	USER	domain.lifecycle	INFO	MWM_VM_MANUAL_MIGRATE_START	VM %s migration started manually. (%s)	The named VM migration has been started manually. Additional information may be provided regarding source and destination nodes.
<b>0x100-0260</b>	USER	domain.lifecycle	INFO	MWM_VM_READY	VM %s is ready.	The named VM is ready. It has been linked to an internal tracking job.
<b>0x100-2711</b>	USER	system-moab	INFO	MWM_PARAMETER_SET_TO_VALUE_INFO	Parameter '%s' is set to '%s'.	A parameter was set to a specified value. This is usually accomplished via a configuration file.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2741</b>	USER	system.-moab	INFO	MWM_SOCKET_EXCEPTION	Exception detected in select for socket %s.	The select() system call indicated an exception for this socket.
<b>0x100-2742</b>	USER	system.-moab	INFO	MWM_SOCKET_EXCEPTION_REASON	Exception identified as '%s' in select for socket %s.	The select() system call indicated an exception for this socket. It has been identified with an error id by getsockopt().
<b>0x100-2748</b>	USER	system.-moab	INFO	MWM_MOAB_STARTED_ON_CORRECT_HOST	Server started on host '%s' %s.	Moab is started on either the primary or fallback server.
<b>0x100-2762</b>	USER	system.-moab	INFO	MWM_CONFIG_LINE_SUCCESSFUL	Configuration line '%s' successfully processed.	The line in the configuration file was processed without error.
<b>0x100-2935</b>	USER	system.-moab	INFO	MWM_ACTIVE_JOB_REMOVED_FROM_QUEUE	Active %s job %s has been removed from the queue, default to successful completion.	The job was removed from the indicated resource manager while it was still active. By default it is assumed to complete successfully unless more information is available (i.e. ENABLEFAILUREFORPURGEDJOB).

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2936</b>	USER	system-moab	INFO	MWM_INACTIVE_JOB_REMOVED_FROM_QUEUE	Inactive %s job %s has been removed from the queue, default to status 'canceled'.	The job was removed from the indicated resource manager while it was still active. By default it is given status 'canceled' unless more information is available (i.e. ENABLEFAILUREFORPURGEDJOB).
<b>0x100-2937</b>	USER	system-moab	INFO	MWM_RM_DOWN_SKIPPING_WORK	RM %s state is %s, skipping %s.	The specified resource manager is not in a good state. Certain actions might be skipped while it is in this state.
<b>0x100-296a</b>	USER	system-moab	INFO	MWM_CANNOT_RESUME_JOB	Cannot resume job '%s' (%s).	Check the PBS server log to see reason for failure.
<b>0x100-296b</b>	USER	system-moab	INFO	MWM_CANNOT_LOCATE_RESOURCE	Cannot locate %s '%s'.	Unable to find the resource specified.
<b>0x100-296c</b>	USER	system-moab	INFO	MWM_CANNOT_SET_JOB_CLASS	Cannot set class on job '%s' to '%s' (%s).	The job could not be modified.
<b>0x100-296d</b>	USER	system-moab	INFO	MWM_NATIVE_ACTION_MISSING	%s action not specified for native interface. %s.	The native interface allows custom actions to be specified. No value was specified for this action.
<b>0x100-296e</b>	USER	system-moab	INFO	MWM_COMMAND_SENT	Command sent to server.	A command was sent.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-296f</b>	USER	system-moab	INFO	MWM_JOB_MAXPREMPT	JOBMAXPREEMPTPE- RITERATION reached: %s of %s.	The maximum value was reached.
<b>0x100-2970</b>	USER	system-moab	INFO	MWM_JOB_CHANGED_STATES	Job '%s' changed states from '%s' to '%s'.	The state changed.
<b>0x100-2971</b>	USER	system-moab	INFO	MWM_JOB_RELEASING_RESERVATION	Job '%s' was requeued/rejected. Releasing reservation.	The job no longer holds the reservation.
<b>0x100-2972</b>	USER	system-moab	INFO	MWM_NODE_CHANGED_STATES	Node '%s' changed states from '%s' to '%s'.	The node state changed.
<b>0x100-2973</b>	USER	system-moab	INFO	MWM_JOB_ACTION_SUCCESSFUL	Job '%s' successfully %s.	The job action completed.
<b>0x100-2974</b>	USER	system-moab	INFO	MWM_ALLOC_TEMP_MEMORY	Cannot allocate temp memory for %s completed jobs.	The system may be low on memory.
<b>0x100-2975</b>	USER	system-moab	INFO	MWM_ACTION_LAUNCHED	Action '%s' launched with message '%s'.	Scheduler action is about to be executed.
<b>0x100-2976</b>	USER	system-moab	INFO	MWM_JOB_ADJUSTMENT	Adjusting allocated %s to %s for job '%s'.	The value is being changed.
<b>0x100-2977</b>	USER	system-moab	INFO	MWM_ALL_JOBS_LOADED	All jobs loaded.	The jobs have been loaded.
<b>0x100-2978</b>	USER	system-moab	INFO	MWM_ALL_NODES_LOADED	All located non-native nodes loaded (%s).	The nodes have been loaded.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2979</b>	USER	system.-moab	INFO	MWM_BACKFILL_POLICY_DISABLED	Backfill policy disabled.	The policy was disabled.
<b>0x100-297a</b>	USER	system.-moab	INFO	MWM_JOB_LOAD	Cannot load job '%s'.	The job failed to load.
<b>0x100-297b</b>	USER	system.-moab	INFO	MWM_CANNOT_CREATE_RSV	Cannot create reservation.	The request to create the given reservation has failed.
<b>0x100-297c</b>	USER	system.-moab	INFO	MWM_MODIFY_PARTITION	Cannot modify partition of running job '%s'.	Must wait until job completes.
<b>0x100-297f</b>	USER	system.-moab	INFO	MWM_CANNOT_CREATE_RSV_IN_PARTITION	Cannot create reservation for job '%s' in partition '%s'.	Job cannot be run on requested partition.
<b>0x100-2982</b>	USER	system.-moab	INFO	MWM_CLUSTER_QUERY_GETDATA	Cluster query get-data failed for native interface.	The resource manager may be down or unresponsive.
<b>0x100-2a0b</b>	USER	system.-moab	INFO	MWM_SENDING_CLIENT_COMMAND	Sending %s command: '%s'.	The specified command is being sent to the server.
<b>0x100-2a0e</b>	USER	system.-moab	INFO	MWM_SCHED_SHUTDOWN_REQUEST	The scheduler has received a user shutdown request.	The Moab scheduler has received a request to shut down. It will be processed as soon as possible.
<b>0x100-2a0f</b>	USER	system.-moab	INFO	MWM_SCHED_RECYCLE_REQUEST	The scheduler has received a user recycle request.	The Moab scheduler has received a request to recycle. It will be processed as soon as possible.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2a10</b>	USER	system-moab	INFO	MWM_SCHED_PAUSE_DESCRIPTION	Scheduling will be disabled, cluster information will continue to be updated.	This is a description of what happens when the scheduler is paused.
<b>0x100-2a11</b>	USER	system-moab	INFO	MWM_SCHED_STOP_TIMESTAMP	Scheduling will stop in %s at iteration %s.	This provides a log message of when the scheduler will stop.
<b>0x100-2a12</b>	USER	system-moab	INFO	MWM_SCHED_RESUME_TIMESTAMP	Scheduling will resume in %s seconds.	This provides a log message of when the scheduler will resume.
<b>0x100-2a13</b>	USER	system-moab	INFO	MWM_SCHED_RESTART_TIME_REACHED	Scheduler restart time reached (scheduler will restart).	The configured restart time was reached. (RESTARTINTERVAL or MEMREFRESHINTERVAL.
<b>0x100-2a14</b>	USER	system-moab	INFO	MWM_SCHED_COMPLETE_SCHEDULING	Scheduling complete. Sleeping for %s seconds.	The scheduling portion of the iteration is complete. Additional jobs will not be scheduled until the next iteration.
<b>0x100-2a17</b>	USER	system-moab	INFO	MWM_ABOUT_TO_EXEC	About to exec() '%s'.	The process is about to be executed.
<b>0x100-2a18</b>	USER	system-moab	INFO	MWM_JOB_ARRAY_CANCEL_POLICY	Sub-job %s exit code %s canceled job array %s with policy %s.	A job within an array job finished and, depending on its exit code and the policy in place, the entire array job might cancel.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2a19</b>	USER	system.-moab	INFO	MWM_RESERVATION_COMPLETION_DELAYED	Reservation completion for job '%s' delayed from %s to %s.	The reservation end time is later than initially expected for this job.
<b>0x100-2a1b</b>	USER	system.-moab	INFO	MWM_VM_ORPHANED	VM '%s' successfully orphaned.	The VM has been separated from its tracking job.
<b>0x100-2a1c</b>	USER	system.-moab	INFO	MWM_VM_REPORTED_DESTROYED	VM '%s' reported destroyed via RM - removing VM.	The VM is no longer available from the resource manager, so it is being removed from the scheduler.
<b>0x100-2a1d</b>	USER	system.-moab	INFO	MWM_VM_STALE_REPORT	VM '%s' has not been reported in %s seconds.	The VM is no longer being reported from the resource manager. No action is currently being taken.
<b>0x100-2a1e</b>	USER	system.-moab	INFO	MWM_WIKI_KEYWORD_NOT_HANDLED	Wiki keyword '%s' (%s) not handled.	The keyword was not recognized, so it will be ignored.
<b>0x100-2a20</b>	USER	system.-moab	INFO	MWM_ROLLING_LOGFILE	Rolling logfile '%s' to '%s'.	The old logfile will be closed and logging will resume in the new file.
<b>0x100-2a23</b>	USER	system.-moab	INFO	MWM_NODE_LOCATED	Nodes located for job %s: %s of %s required (%s feasible).	List of nodes located for a specific job.
<b>0x100-2a2d</b>	USER	system.-moab	INFO	MWM_JOB_PAL_SET	Partition access list set to value: %s.	The partition access list (PAL) is set.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2a2e</b>	USER	system-moab	INFO	MWM_JOB_PREEMPTED_BY_JOB	Job %s preempted job %s - added idle resources (T: %s; N: %s; P: %s)/remaining (T: %s; N: %s; P: %s).	Job was preempted by another job.
<b>0x100-2a2f</b>	USER	system-moab	INFO	MWM_JOB_CAN_START_WITHOUT_PREEMPTION	Job %s would start in %s without preemption (PC: %s).	Job can start without preemption.
<b>0x100-2a32</b>	USER	system-moab	INFO	MWM_PARTITION_RESOURCES	Partition %s nodes/-procs available %s/%s (%s jobs examined).	General partition information.
<b>0x100-2a33</b>	USER	system-moab	INFO	MWM_RSV_OPERATION	Performing '%s' operation on reservation expression '%s' (%s matches).	This is operation is caused by a mrsvctl command.
<b>0x100-2a34</b>	USER	system-moab	INFO	MWM_PREEMPTING_JOBS	Preempting jobs to allow job %s to start - required resources T: %s; N: %s; P: %s.	Preempting jobs.
<b>0x100-2a35</b>	USER	system-moab	INFO	MWM_MOABTRACKSUSPEND	Preempt usage tracking enabled (env).	Environment variable MOABTRACKSUSPEND set.
<b>0x100-2a36</b>	USER	system-moab	INFO	MWM_JOB_MAX_PREEMPTEE_LIMIT	Single job max preemptee limit (%s) reached.	Max requirements exceeded on job.
<b>0x100-2a37</b>	USER	system-moab	INFO	MWM_QUEUES_DETECTED	Queues detected: %s.	Resource manager found queues on cluster query update.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2a38</b>	USER	system.-moab	INFO	MWM_JOB_START_TIME_CHANGED	Start time changed from %s to %s on job %s.	The job's start time was changed via the resource manager.
<b>0x100-2a39</b>	USER	system.-moab	INFO	MWM_STORING_CHECKPOINT_INFO	Storing object to checkpoint.	The object's state is being checkpointed.
<b>0x100-2a3a</b>	USER	system.-moab	INFO	MWM_PBS_DATA_UP_TO_DATE	PBS raw data already up to date.	The resource manager is already updated.
<b>0x100-2a3b</b>	USER	system.-moab	INFO	MWM_PBS_DATA_UPDATED	PBS data updated for iteration %s.	The resource manager is now updated.
<b>0x100-2a3c</b>	USER	system.-moab	INFO	MWM_STARTED_MESSAGE_QUEUE	Started message queue thread.	The message queue is now operational.
<b>0x100-2a3d</b>	USER	system.-moab	INFO	MWM_JOBS_SELECTED_IN_PARTITION	Total jobs selected in partition %s: %s/%s.	Identifies the selected jobs in a partition.
<b>0x100-2a3e</b>	USER	system.-moab	INFO	MWM_TASKS_LOCATED_FOR_JOB	Tasks located for job %s: %s of %s required (%s feasible).	Identifies the tasks available for a job.
<b>0x100-2a3f</b>	USER	system.-moab	INFO	MWM_CLIENT_REQUEST	Client requesting command '%s'.	Client requested command.
<b>0x100-2a40</b>	USER	system.-moab	INFO	MWM_REQUEST_TO_CANCEL_JOB	Request to cancel job '%s' sent, but could not confirm cancellation (pending response).	Client did not get a confirmation as expected.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-2a41</b>	USER	system-moab	INFO	MWM_RESERVATION_NOT_ALLOWED_FOR_JOB	Reservation not allowed for job %s in %s.	Reservation not allowed in specified condition.
<b>0x100-2a42</b>	USER	system-moab	INFO	MWM_RESERVED_JOB_STARTED	Reserved job '%s' started.	Reserved job started.
<b>0x100-2a43</b>	USER	system-moab	INFO	MWM_RESOURCES_AVAILABLE_AFTER_SCHEDULING	Resources available after scheduling: N: %s P: %s.	Resources available after scheduling.
<b>0x100-2a44</b>	USER	system-moab	INFO	MWM_RESTORING_DEFERRED_JOB	Restoring job '%s' from deferred state.	Restoring job from deferred state.
<b>0x100-2a45</b>	USER	system-moab	INFO	MWM_RM_DUPLICATE_QUERY	RM %s already has a pending query - skipping get data query.	Duplicate queries may not be performed simultaneously.
<b>0x100-2a46</b>	USER	system-moab	INFO	MWM_RM_PEER_COMMAND	Sending peer server command to %s:%s (Cmd: %s, Requestor: %s, Key: %s...).	A command has been sent to a peer Moab grid server.
<b>0x100-2a47</b>	USER	system-moab	INFO	MWM_SET_ATTRIBUTE_ON_NODE	Setting %s on node %s to %s.	A command has been sent to a peer Moab grid server.
<b>0x100-2a48</b>	USER	system-moab	INFO	MWM_SET_ATTRIBUTE_ON_JOB	Setting %s on job %s to %s (%s).	A command has been sent to a peer Moab grid server.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a73c</b>	ADMIN	system-moab	INFO	MWM_CANNOT_STAT_FILE_INFO	Cannot stat file '%s', errno: %s (%s).	The stat() system call failed. This is not always significant as it is sometimes used to test the existence of a file that may or may not be there. Use the errno and associated message to determine possible causes.
<b>0x100-a743</b>	ADMIN	system-moab	INFO	MWM_FAILED_SELECT	Select for socket %s failed, errno: %s (%s).	The select() system call failed. Use the errno and associated message to determine possible causes.
<b>0x100-a744</b>	ADMIN	system-moab	INFO	MWM_SELECT_TIMEOUT	Select for socket %s timed out after %s seconds with no valid descriptors.	The select() system call timed out. This may or may not be an error. Check MTU.
<b>0x100-a75d</b>	ADMIN	system-moab	INFO	MWM_CONFIG_VALUE_OUT_OF_RANGE	Configuration parameter '%s' has an invalid value '%s'. Range is limited by %s.	Check the line in the configuration file for the attribute.
<b>0x100-a76d</b>	ADMIN	system-moab	INFO	MWM_REMOVING_OBJECT_FROM_MONGO	Removing object '%s' from Mongo DB '%s'.	The object is being removed from the database.
<b>0x100-a789</b>	ADMIN	system-moab	INFO	MWM_UNABLE_TO_ALLOCATE_NODES_FOR_RSV	Cannot allocate nodes for reservation '%s'. (%s)	Cannot allocate a node list that matches the requirements for this reservation. This may not be serious since multiple passes may occur.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a8ab</b>	ADMIN	system-moab	INFO	MWM_VM_EXCEED_TTL	VM '%s' has reached TTL (%s). Must be removed manually.	The given VM has reached its time to live.
<b>0x100-a8e1</b>	ADMIN	system-moab	INFO	MWM_PLUGIN_LOADED_SUCCESS	Successfully loaded NodeAllocation plugin '%s' for partition '%s'.	A NodeAllocation plugin was loaded without error.
<b>0x100-a95b</b>	ADMIN	system-moab	INFO	MWM_RSV_FULL	Full reservation '%s' reserved %s procs in partition '%s' to start in %s at (%s).	The full reservation has been reserved.
<b>0x100-a97d</b>	ADMIN	system-moab	INFO	MWM_RSV_PREREQ_JOB	Cannot create reservation for prerequisite job '%s'.	Could not obtain a reservation for this job.
<b>0x100-a97e</b>	ADMIN	system-moab	INFO	MWM_ANNOTATE_JOB	Cannot annotate job '%s' with message '%s'.	Unable to modify the job with the annotation.
<b>0x100-a980</b>	ADMIN	system-moab	INFO	MWM_UPDATE_JOB	Cannot update job '%s'.	The update on the job from XML failed.
<b>0x100-a981</b>	ADMIN	system-moab	INFO	MWM_REMAP_CLASS	Cannot remap class for RM job '%s' (%s).	Unable to modify the job with the new class.
<b>0x100-a983</b>	ADMIN	system-moab	INFO	MWM_COMPLETING_JOB	Completing job '%s'.	The job finished.
<b>0x100-a984</b>	ADMIN	system-moab	INFO	MWM_JOBS_DETECTED	There were %s %s jobs detected on RM '%s'.	The resource manager reported these jobs.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a985</b>	ADMIN	system-moab	INFO	MWM_SUSPEND_JOB	Cannot suspend job '%s' (%s).	Check the PBS server log to see reason of failure.
<b>0x100-a986</b>	ADMIN	system-moab	INFO	MWM_STALE_PARTITION	Attempting to remove stale partition for completed job '%s'.	About to perform the stated operation.
<b>0x100-a987</b>	ADMIN	system-moab	INFO	MWM_STALE_PARTITION_SUCCESS	Successfully removed stale partition for completed job '%s'.	Successfully performed the stated operation.
<b>0x100-a988</b>	ADMIN	system-moab	INFO	MWM_CANCEL_NOQUEUE_JOB	Canceling No-queue job '%s'.	About to perform the stated operation.
<b>0x100-a989</b>	ADMIN	system-moab	INFO	MWM_SIGNAL_JOB	Cannot signal job '%s' (%s).	The resource manager did not respond to the signal request.
<b>0x100-a98a</b>	ADMIN	system-moab	INFO	MWM_RSV_JOB_CREDS	Cannot set up reservation job credentials.	The user, account, or group credentials may not be valid.
<b>0x100-a98b</b>	ADMIN	system-moab	INFO	MWM_CP_CORRUPT_NODE_LINE	Corrupt node line detected (%s).	The line does not contain the correct syntax for a checkpoint.
<b>0x100-a98c</b>	ADMIN	system-moab	INFO	MWM_EVALUATING_RSV	Evaluating reservation '%s'.	About to perform the stated operation.
<b>0x100-a98d</b>	ADMIN	system-moab	INFO	MWM_EXPIRING_CHECKPOINT_DATA	Expiring checkpoint data for %s '%s'. Not updated in %s.	The object's checkpoint data has expired.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a98e</b>	ADMIN	system-moab	INFO	MWM_JOB_PREVIOUSLY_REMOVED	Job '%s' was previously removed.	The job has already been removed.
<b>0x100-a98f</b>	ADMIN	system-moab	INFO	MWM_JOB_STARTED_BY_USER	Job '%s' was started by user '%s'.	The job started.
<b>0x100-a990</b>	ADMIN	system-moab	INFO	MWM_JOB_NOT_STARTED_BY_USER	Job '%s' could not be started by user '%s' (%s).	The job could not be started.
<b>0x100-a991</b>	ADMIN	system-moab	INFO	MWM_RM_JOB_NOT_STARTED	Job '%s' could not be started with %s RM '%s' (%s).	The job could not be started.
<b>0x100-a992</b>	ADMIN	system-moab	INFO	MWM_JOB_CANCELED_EXTERNALLY	Job '%s' appears to have been canceled externally.	The job was canceled.
<b>0x100-a993</b>	ADMIN	system-moab	INFO	MWM_JOB_COMPLETED_SINGLE_ITERATION	Job '%s' appears to have been started and completed in a single iteration.	The job completed.
<b>0x100-a994</b>	ADMIN	system-moab	INFO	MWM_JOB_PROCESSING_COMPLETED	Job processing completed.	The jobs have been processed.
<b>0x100-a995</b>	ADMIN	system-moab	INFO	MWM_PROCESSING_JOB	Processing job '%s' in state '%s'.	Processing a single job.
<b>0x100-a996</b>	ADMIN	system-moab	INFO	MWM_JOB_SUSPENDED	Job '%s' suspended through %s RM.	The resource manager suspended the job.
<b>0x100-a997</b>	ADMIN	system-moab	INFO	MWM_JOB_RESUMED	Job '%s' resumed through %s RM.	The resource manager resumed the job.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a998</b>	ADMIN	system-moab	INFO	MWM_JOB_FEASIBILITY_CHECK_DISABLED	Job feasibility check disabled (env).	This feature has been disabled.
<b>0x100-a999</b>	ADMIN	system-moab	INFO	MWM_JOB_USAGE_SENT	Job usage sent for job '%s'.	The usage sent as feedback to user.
<b>0x100-a99a</b>	ADMIN	system-moab	INFO	MWM_LOADING_JOBS	Loading %s job(s).	The jobs are about to be loaded.
<b>0x100-a99b</b>	ADMIN	system-moab	INFO	MWM_LOADING_NODE_RECORDS	Loading %s node record(s).	The node records are about to be loaded.
<b>0x100-a99c</b>	ADMIN	system-moab	INFO	MWM_LOADED_WORKLOAD_BUFFER	Loaded %s workload buffer (%s bytes), processing jobs.	The workload buffer was loaded.
<b>0x100-a99d</b>	ADMIN	system-moab	INFO	MWM_JOB_REJECTED_INFINITE_WALLTIME	Job '%s' rejected (requested infinite walltime).	Jobs must have a walltime limit.
<b>0x100-a99e</b>	ADMIN	system-moab	INFO	MWM_JOB_REJECTED_PARTITION	Job '%s' rejected in partition %s (exceeds maximum task size: %s > %s).	Adjust <code>JOBMAXTASKCOUNT</code> in the configuration file.
<b>0x100-a99f</b>	ADMIN	system-moab	INFO	MWM_JOB_ALREADY_EXISTS	Job '%s' already exists but is not a duplicate.	The ID of the job matched a completed job.
<b>0x100-a9a0</b>	ADMIN	system-moab	INFO	MWM_JOB_ALREADY_BATCH_HOLD	Job '%s' is already on batch hold.	Trying to place a job on hold that is already in that state.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
0x100-a9a1	ADMIN	system-moab	INFO	MWM_JOB_REQUESTS_RSV	Job '%s' requests reservation '%s' (not deferring).	The job is requesting the reservation.
0x100-a9a2	ADMIN	system-moab	INFO	MWM_RM_CONNECTION_FAILED	Connection to RM '%s' failed. Not deferring job '%s' (Reason: %s).	Refer to the reason message.
0x100-a9a3	ADMIN	system-moab	INFO	MWM_DEFER_DISABLED	Defer disabled.	The job cannot be deferred.
0x100-a9a4	ADMIN	system-moab	INFO	MWM_MWS_CLUSTER_QUERY	Cluster query retrieval failed for MWS RM '%s'.	The resource manager did not respond to the request.
0x100-a9a5	ADMIN	system-moab	INFO	MWM_JOB_INVALID_PARTITION	Job '%s' specifies an invalid partition.	The job must reference a valid partition.
0x100-a9a6	ADMIN	system-moab	INFO	MWM_JOB_INVALID_QoS	Cannot set QoS on job '%s' to '%s' - invalid QoS.	The job must use a valid QoS.
0x100-a9a7	ADMIN	system-moab	INFO	MWM_JOB_INVALID_ACCOUNT	Cannot set account on job '%s' to '%s' - invalid account (%s).	The job must use a valid account.
0x100-a9a8	ADMIN	system-moab	INFO	MWM_CHECKING_IDLE_JOB	Checking idle job '%s' (priority: %s) partition %s.	Checking the job.
0x100-a9a9	ADMIN	system-moab	INFO	MWM_CHECKING_SUSPENDED_JOB	Checking suspended job '%s' (priority: %s).	Checking the job.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9aa</b>	ADMIN	system-moab	INFO	MWM_CHECKPOINT_TEST_ENABLED	Checkpoint test enabled (env).	The feature has been enabled with an environment variable.
<b>0x100-a9ab</b>	ADMIN	system-moab	INFO	MWM_ADD_NODE_FAILED	Could not add node because MNodeAdd failed.	The node could not be added to the object.
<b>0x100-a9ac</b>	ADMIN	system-moab	INFO	MWM_ATTEMPTING_RESERVATION	Attempting reservation of %s procs in %s for %s.	The scheduler will try to make the reservation.
<b>0x100-a9ad</b>	ADMIN	system-moab	INFO	MWM_FAIRSHARE_INTERVAL	Fairshare rolled to interval %s.	The interval has changed.
<b>0x100-a9ae</b>	ADMIN	system-moab	INFO	MWM_INVALID_ARCHITECTURE	Invalid architecture.	The architecture is not a valid value.
<b>0x100-a9af</b>	ADMIN	system-moab	INFO	MWM_INVALID_PSEUDOJOB	Invalid pseudo-job.	The pseudo-job is not a valid value.
<b>0x100-a9b0</b>	ADMIN	system-moab	INFO	MWM_HOLD_TYPE	Hold type '%s' selected.	The given hold type was specified.
<b>0x100-a9b1</b>	ADMIN	system-moab	INFO	MWM_MESSAGE_SENT	Message sent to server.	The message was sent.
<b>0x100-a9b2</b>	ADMIN	system-moab	INFO	MWM_JOB_LOCATED	Located job '%s' in partition '%s' reserved to start %s.	The specified job has been located.
<b>0x100-a9b3</b>	ADMIN	system-moab	INFO	MWM_TOTAL_JOBS_DETECTED	Total jobs detected: %s.	Number of counted jobs returned from the workload query.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9b4</b>	ADMIN	system-moab	INFO	MWM_NO_WORKLOAD_DETECTED	No workload reported by any RM.	No jobs were reported across all the resource manager queries.
<b>0x100-a9b5</b>	ADMIN	system-moab	INFO	MWM_LOADING_JOB	Loading job '%s' in state '%s' (%s bytes).	The job is being loaded.
<b>0x100-a9b6</b>	ADMIN	system-moab	INFO	MWM_JOB_START_REJECTED	Local constraints rejected the starting of job '%s'.	The job cannot start.
<b>0x100-a9b7</b>	ADMIN	system-moab	INFO	MWM_INVALID_STAT_TYPE	Invalid stat type '%s' requested.	Not a valid value.
<b>0x100-a9b8</b>	ADMIN	system-moab	INFO	MWM_ORPHAN_PARTITION	Creating temporary job to process orphan partition '%s' for job '%s'.	The job was not found in active or completed job tables.
<b>0x100-a9b9</b>	ADMIN	system-moab	INFO	MWM_DISABLING_ACTION_PROGRAM	Disabling action program '%s'.	An invalid action program was requested.
<b>0x100-a9ba</b>	ADMIN	system-moab	INFO	MWM_DISABLING_JOB_FB_PROGRAM	Disabling job feedback program '%s' (%s).	An invalid job feedback program was requested. See documentation for FEEDBACKPROGRAM.
<b>0x100-a9bb</b>	ADMIN	system-moab	INFO	MWM_CP_RESTART_STATE_IGNORED	Checkpoint restart state '%s' ignored.	The restart state specified is being ignored.
<b>0x100-a9bc</b>	ADMIN	system-moab	INFO	MWM_CP_RESTART_STATE_SUCCESS	Starting scheduler with checkpoint restart state '%s'.	The restart state specified is being used.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9bd</b>	ADMIN	system.-moab	INFO	MWM_DESTROYING_NODE	Destroying node '%s'.	The specified node is being destroyed.
<b>0x100-a9be</b>	ADMIN	system.-moab	INFO	MWM_IGNOREING_NODE	Ignoring node '%s'.	The specified node is being ignored.
<b>0x100-a9c0</b>	ADMIN	system.-moab	INFO	MWM_CANNOT_ADJUST_JOB_HOLDS	Cannot adjust holds on remote peer for job '%s' (%s).	Unable to modify the job.
<b>0x100-a9c1</b>	ADMIN	system.-moab	INFO	MWM_JOB_CANNOT_CREATE_RESERVATION	Cannot create reservation for job '%s' (previously reserved to start in %s)).	Failed to create reservation for job.
<b>0x100-a9c2</b>	ADMIN	system.-moab	INFO	MWM_TRIGGER_LOAD_OUTPUT	Cannot load output data for trigger '%s' (File: %s).	The file may not exist or may be inaccessible.
<b>0x100-a9c3</b>	ADMIN	system.-moab	INFO	MWM_PBS_SERVER_CONNECT	Connected to PBS server %s:%s on sd %s.	Connection established.
<b>0x100-a9c4</b>	ADMIN	system.-moab	INFO	MWM_NO_JOB_DATA	No job data was sent by %s RM.	The data sent by the resource manager did not contain job information.
<b>0x100-a9c5</b>	ADMIN	system.-moab	INFO	MWM_JOB_RESUMED_WITH_PROCS	Job '%s' resumed on %s processors.	The resource manager resumed the job.
<b>0x100-a9c6</b>	ADMIN	system.-moab	INFO	MWM_JOB_SINGALED	Job %s' successfully signaled (action: %s, signal: %s).	The job responded to the signal request.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9c7</b>	ADMIN	system-moab	INFO	MWM_JOB_CANCELED_RM	Job '%s' canceled through %s RM.	The job was canceled.
<b>0x100-a9c8</b>	ADMIN	system-moab	INFO	MWM_JOB_ASSIGNED_DEFAULT_GROUP	Job '%s' assigned default group '%s'.	The job was modified.
<b>0x100-a9c9</b>	ADMIN	system-moab	INFO	MWM_FILE_EXECUTE_PERMISSION	File '%s' does not have user execute permission (st_mode = %s).	The permissions must be modified.
<b>0x100-a9ca</b>	ADMIN	system-moab	INFO	MWM_INSUFFICIENT_PREEMPT_JOBS	Inadequate preempt jobs (%s) located for %s job (P: %s of %s, N: %s of %s).	Not enough jobs could be preempted.
<b>0x100-a9cb</b>	ADMIN	system-moab	INFO	MWM_READ_STAT_INDEX	Cannot read stat index for location %s:%s:%s.	The checkpoint did not have the stat information.
<b>0x100-a9cc</b>	ADMIN	system-moab	INFO	MWM_BACKFULL_JOB_PREEMPT	Backfill job '%s' no longer preemptible (%s > %s) in partition '%s'.	The job cannot be preempted.
<b>0x100-a9cd</b>	ADMIN	system-moab	INFO	MWM_STARTTIME_UNAVAILABLE	Cannot obtain desired starttime (%s != %s).	The job cannot be adjusted to the given start time.
<b>0x100-a9ce</b>	ADMIN	system-moab	INFO	MWM_STARTTIME_ADJUSTED	Timeframe for reservation %s adjusted forward by %s seconds.	The reservation has been adjusted to the given start time.
<b>0x100-a9cf</b>	ADMIN	system-moab	INFO	MWM_RESERVATION_ROLLBACK	Time: %s Roll-backOffset: %s RsvStartTime: %s RsvDuration %s.	The reservation is being considered for rollback.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9d0</b>	ADMIN	system-moab	INFO	MWM_RESERVATION_NOT_REQUIRED	Reservation '%s' not required for specified period.	The reservation is not required for this time period.
<b>0x100-a9d1</b>	ADMIN	system-moab	INFO	MWM_RM_INTERFACE_RECOVERED	The interface for RM '%s' has been recovered.	A previously corrupt interface is now working.
<b>0x100-a9d2</b>	ADMIN	system-moab	INFO	MWM_NTR_JOB_FOUND	Found an NTR (next to run) job - stopping idle job scheduling.	The job will now be run.
<b>0x100-a9d3</b>	ADMIN	system-moab	INFO	MWM_GRES_KEYWORD_NO_VALUE	GRes keyword '%s' passed in with no value.	A value must be specified.
<b>0x100-a9d4</b>	ADMIN	system-moab	INFO	MWM_JOB_EXTENSION_STRING	Job '%s' has invalid extension string - '%s'.	The system is unable to process the string.
<b>0x100-a9d5</b>	ADMIN	system-moab	INFO	MWM_JOB_PROCESS_FAILURE	Job '%s' is invalid. It cannot be processed (%s).	There was an error loading the job. It will be rejected.
<b>0x100-a9d6</b>	ADMIN	system-moab	INFO	MWM_JOB_MODIFIED_RM	Job '%s' has been modified through %s RM.	The job was modified.
<b>0x100-a9d7</b>	ADMIN	system-moab	INFO	MWM_IDLE_BACKLOG_SIZE	Idle backlog: %s seconds (%s hours).	The idle backlog status is given.
<b>0x100-a9d8</b>	ADMIN	system-moab	INFO	MWM_SET_RESOURCES	Inadequate resources found in any set (%s < %s).	None of the node sets have the resources needed.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9d9</b>	ADMIN	system-moab	INFO	MWM_JOB_PREEMPTING_JOB	Job %s preempting job %s (statemtime: %s) (preempted this iteration: %s).	One job preempted another.
<b>0x100-a9da</b>	ADMIN	system-moab	INFO	MWM_UPDATE_SCHEDULER_STATS	Iteration: %s; scheduling time: %s seconds.	Normal statistics update.
<b>0x100-a9db</b>	ADMIN	system-moab	INFO	MWM_JOB_STARTED_RM	Job '%s' started through %s RM on %s procs.	The job has started.
<b>0x100-a9dc</b>	ADMIN	system-moab	INFO	MWM_JOB_DELAY	Job delay: %s; reservation retry time: %s (StateDelayNC: %s; JobRsvDelayNC: %s).	The job has been delayed.
<b>0x100-a9dd</b>	ADMIN	system-moab	INFO	MWM_JOB_COMPLETED	Job '%s' completed. X: %s; T: %s; PS: %s; A: %s (RM: %s/%s).	The job completed.
<b>0x100-a9de</b>	ADMIN	system-moab	INFO	MWM_JOB_RESERVED_TASKS	Job '%s' reserved %s tasks (partition %s) to start in %s on %s (WC: %s).	The job has reserved the tasks.
<b>0x100-a9df</b>	ADMIN	system-moab	INFO	MWM_EVENT_INTERFACE_ENABLED	Event interface enabled for wiki RM %s on port %s.	The interface is now functional.
<b>0x100-a9e0</b>	ADMIN	system-moab	INFO	MWM_RM_RESOURCES_DETECTED	There were %s %s resources detected on RM '%s'.	The given resources were found.
<b>0x100-a9e3</b>	ADMIN	system-moab	INFO	MWM_EXTENDING_RESERVATION	Extending reservation by %s seconds (trigger still active).	The reservation is being extended.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9e4</b>	ADMIN	system.-moab	INFO	MWM_EXTENDING_RESERVATION_OVERRUN_JOB	Extending reservation for overrun job '%s' by %s seconds.	The reservation is being extended.
<b>0x100-a9e5</b>	ADMIN	system.-moab	INFO	MWM_JOB_LOCATED_BESTFIT	Located bestfit job '%s' (size: %s; duration: %s).	Backfill found a job that best fits the available resources.
<b>0x100-a9e6</b>	ADMIN	system.-moab	INFO	MWM_JOB_BEST_PARTITION	The best partition for job '%s' is '%s'.	Backfill found a job that best fits the available resources.
<b>0x100-a9e7</b>	ADMIN	system.-moab	INFO	MWM_CPA_PARTITION_DESTROY	Destroying CPA partition '%s' for job '%s' with cookie %s (%s).	The partition is being destroyed.
<b>0x100-a9e8</b>	ADMIN	system.-moab	INFO	MWM_RESOURCES_LOCATED	Located resources for %s tasks (%s) in best partition '%s' for job '%s' at time offset %s.	The listed resources have been located.
<b>0x100-a9e9</b>	ADMIN	system.-moab	INFO	MWM_MINIMUM_EFFICIENCY_REACHED	Minimum efficiency reached (%s percent) on iteration %s.	The threshold has been reached.
<b>0x100-a9ea</b>	ADMIN	system.-moab	INFO	MWM_JOB_START_PARTITION	Cannot start job '%s' in partition '%s' (scheduler mode: %s).	The job could not be started.
<b>0x100-a9eb</b>	ADMIN	system.-moab	INFO	MWM_JOB_FEASIBLE_NODES	Inadequate feasible nodes found for job '%s':%s in partition '%s' (%s < %s).	The job could not be scheduled.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9ec</b>	ADMIN	system-moab	INFO	MWM_JOB_LOADED	Job '%s' loaded: TC=%s UGC-C=%s,%s,%s WC=%s ST=%s %s %s.	The job was loaded.
<b>0x100-a9ed</b>	ADMIN	system-moab	INFO	MWM_JOB_RSV_CREATE	Cannot create new reservation for job %s (shape[%s] %s).	Check the reservation time, nodes, and account.
<b>0x100-a9ee</b>	ADMIN	system-moab	INFO	MWM_JOB_LOCATE_NODES	Cannot locate nodes for job '%s' req[%s] (%s additional needed).	Not enough nodes are available to run the job.
<b>0x100-a9ef</b>	ADMIN	system-moab	INFO	MWM_JOB_START_RM_DISABLED	Cannot start job '%s' since RM '%s' is disabled.	Not enough nodes are available to run the job.
<b>0x100-a9f0</b>	ADMIN	system-moab	INFO	MWM_JOB_START_RESERVE_TIME	Cannot start job '%s' reserve time in %s.	The time to schedule has already arrived.
<b>0x100-a9f1</b>	ADMIN	system-moab	INFO	MWM_ERROR_IN_EXE_STDERR	Error detected in '%s' due to presence of the word 'ERROR' in stderr (%s).	The executable failed.
<b>0x100-a9f2</b>	ADMIN	system-moab	INFO	MWM_ERROR_IN_STDERR	Error detected due to presence of the word 'ERROR' in stderr.	The child process failed.
<b>0x100-a9f3</b>	ADMIN	system-moab	INFO	MWM_CHECKJOB_STATE	Job '%s' State: %s Expected State: %s QueueTime: %s.	The job is in the listed state. The expected state may not be the same.
<b>0x100-a9f4</b>	ADMIN	system-moab	INFO	MWM_JOB_NODELIST	Cannot obtain nodelist for job '%s':%s in range %s.	The nodes are not available.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9f5</b>	ADMIN	system-moab	INFO	MWM_JOB_RESUME	Job '%s' cannot be resumed since allocated nodes are not available (node '%s' state '%s').	The resource manager resumed the job.
<b>0x100-a9f6</b>	ADMIN	system-moab	INFO	MWM_CLEARING_EXPIRED_RESERVATION	Clearing expired %s reservation '%s' on iteration %s (start: %s end: %s).	The reservation has expired.
<b>0x100-a9f7</b>	ADMIN	system-moab	INFO	MWM_CPA_RETRY	CPA retry detected - will re-attempt partition creation in 2 seconds.	The partition may be created.
<b>0x100-a9f9</b>	ADMIN	system-moab	INFO	MWM_JOBS_STARTED	There were %s %s jobs started in partition '%s' on iteration %s.	The jobs were started.
<b>0x100-a9fa</b>	ADMIN	system-moab	INFO	MWM_TASKS_ALLOCATED	There were %s of %s tasks allocated for job '%s':%s.	The tasks were allocated.
<b>0x100-a9fb</b>	ADMIN	system-moab	INFO	MWM_CLASSES_DETECTED	There were %s %s classes/queues detected on RM '%s'.	The classes were detected.
<b>0x100-a9fd</b>	ADMIN	system-moab	INFO	MWM_JOB_DELAYED_RSV	Delayed reservation detected for reserved job '%s' (%s seconds) attempting squeeze.	Attempting to fit the job into the reservation.
<b>0x100-a9fe</b>	ADMIN	system-moab	INFO	MWM_DUMPING_RESERVATIONS	Dumping reservations on iteration %s.	All the reservations will be dumped to the log.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-a9ff</b>	ADMIN	system-moab	INFO	MWM_ALLOCPARTITION_MISSING	ALLOCPARTITION missing from completed job '%s' - restoring variable with value '%s'.	The value is being substituted.
<b>0x100-aa00</b>	ADMIN	system-moab	INFO	MWM_RECEIVED_NODELIST	Received nodelist through %s RM.	The nodelist was received.
<b>0x100-aa01</b>	ADMIN	system-moab	INFO	MWM_SERVICE_REQUEST_FROM_HOST	Received service request from host '%s'.	The request was received.
<b>0x100-aa02</b>	ADMIN	system-moab	INFO	MWM_RECEIVED_WORKLOAD	Received workload info through %s RM '%s' (%s bytes).	The workload was received.
<b>0x100-aa03</b>	ADMIN	system-moab	INFO	MWM_RSV_REMOVED_FROM_CACHE	Removing reservation '%s' from cache.	The cached reservation is being removed.
<b>0x100-aa04</b>	ADMIN	system-moab	INFO	MWM_RECOVER_READ_SOCKET	RECOVER: attempting to read socket connection.	The recovery function is attempting to communicate via sockets.
<b>0x100-aa05</b>	ADMIN	system-moab	INFO	MWM_GREEDY_BACKFILL	Improved list found by greedy backfill in %s searches (utility: %s; processors available: %s).	The object is being removed from the database.
<b>0x100-aa06</b>	ADMIN	system-moab	INFO	MWM_RESERVATION_NAME_AND_GROUP	Name='%s' RsvGroup='%s'.	The object is being removed from the database.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-aa15</b>	ADMIN	system-moab	INFO	MWM_TRIGGERS_DISABLED	Triggers disabled. %s.	Triggers are disabled. This message indicates when this flag is being set, and when an action is being skipped because the flag is set.
<b>0x100-aa16</b>	ADMIN	system-moab	INFO	MWM_USER_NOT_AUTHORIZED	User %s is not authorized to %s.	This user does not have permissions to accomplish the listed task.
<b>0x100-aa1f</b>	ADMIN	system-moab	INFO	MWM_SUCCESSFULLY_OPENED_SOCKET	Opened service socket on port %s.	A socket was successfully opened listening on the remote port.
<b>0x100-aa21</b>	ADMIN	system-moab	INFO	MWM_NO_CHECKPOINT_INFO	No checkpoint information available for '%s'.	Checkpoint information was not available.
<b>0x100-aa22</b>	ADMIN	system-moab	INFO	MWM_NODE_INDEX_TABLE_ENABLED	Node index table enabled.	Enabled by environment variable: MOABUSENODEINDEX.
<b>0x100-aa24</b>	ADMIN	system-moab	INFO	MWM_UNKNOWN_NODE_SLOT	Node slot not yet set on node '%s'.	Delaying setting rack until slot is known.
<b>0x100-aa25</b>	ADMIN	system-moab	INFO	MWM_NODE_STATUS	Node '%s' status: state='%s' rsvlist='%s' joblist='%s'.	General node status.
<b>0x100-aa26</b>	ADMIN	system-moab	INFO	MWM_NO_JOBS_IN_QUEUE	No jobs in queue.	There were no jobs in the scheduler queue indicating the scheduler has nothing to process.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-aa27</b>	ADMIN	system-moab	INFO	MWM_NO_NODE_DATA	No node data sent by %s RM.	The resource manager did not receive any node data in cluster query.
<b>0x100-aa28</b>	ADMIN	system-moab	INFO	MWM_NO_PREEMPTIBLE_RESOURCES	No preemptible resources found for job %s (tc: %s; class: '%s'; qos: %s; priority: %s; partition %s).	Indicates the scheduler could not find any jobs for preemption.
<b>0x100-aa29</b>	ADMIN	system-moab	INFO	MWM_NO_PRIORITY_RESERVATION_CREATED_FOR_POLICY	No priority reservations created for policy '%s' for job %s.	Job reservation for a specific policy was unable to be created.
<b>0x100-aa2a</b>	ADMIN	system-moab	INFO	MWM_NO_QUEUES_DETECTED	No queues detected.	Resource manager attempted to obtain queue information. Check resource manager for configured queues.
<b>0x100-aa2b</b>	ADMIN	system-moab	INFO	MWM_NOT_ADDING_RM	Not adding RM '%s'.	The partition is not adding the specified resource manager. This situation is most common in grid configurations where resource manager names are similar.
<b>0x100-aa30</b>	ADMIN	system-moab	INFO	MWM_ORPHAN_PARTITION_REPORTED_FOR_JOB	Orphan partition %s reported for job %s.	The resource manager reported the partition as orphaned.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x100-aa31</b>	ADMIN	system-moab	INFO	MWM_PARAMETER_CANNOT_BE_CHANGED	Parameter '%s' cannot be changed while Moab is running.	Configuration file must be changed, and Moab must be restarted.
<b>0x100-e72f</b>	INTERNAL	system-moab	INFO	MWM_SOCKET_REMOTE_DISCONNECT	Reading from a socket failed. It appears the client disconnected, errno: %s (%s).	The recv() system call failed. Use the errno and associated message to determine possible causes.
<b>0x100-e7f0</b>	INTERNAL	system-moab	INFO	MWM_VM_LINKED_TO_NEW_TRACKING_JOB	Setting VMTracking job for VM '%s' to job '%s'.	A VM is associated with a tracking job.
<b>0x110-001f7</b>	USER	domain.lifecycle	WARNING	MWM_TRIG_FAILURE	Trigger %s has failed.	The named trigger has finished its action, but it returned with a failure status.
<b>0x110-0025e</b>	USER	domain.lifecycle	WARNING	MWM_VM_MIGRATE_END_ERROR	VM %s migration has finished with an error: (%s)	The named VM has finished its migration. There was a problem during the migration. Additional information may be provided regarding the error specifics.
<b>0x110-00261</b>	USER	domain.lifecycle	WARNING	MWM_VM_MIGRATE_SUBMIT	Failed to submit VM migration job for VM %s. (%s)	Failed to submit VM Migration job.
<b>0x110-00262</b>	USER	domain.lifecycle	WARNING	MWM_VM_NO_FEASIBLE_NODES	Failed to find a feasible node/hypervisor on which to run VM %s. Check setup job %s for details.	The named VM has been submitted, but no node/hypervisor meets all requirements.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-00384</b>	USER	system-moab	WARN	MWM_VM_LICENSE_ERROR	There is an error with the Moab license: (%s)	There was a licensing error. Additional information may be provided regarding the error specifics.
<b>0x110-0284a</b>	USER	system-moab	WARN	MWM_BAD_COMMANDLINE_FLAG	Unexpected flag detected: '%s'.	The command line syntax that was received contains an invalid flag. Check the documentation and retry.
<b>0x110-02858</b>	USER	system-moab	WARN	MWM_NODESETMAXUSAGE_FAILURE	Ignoring incorrect NODESETMAXUSAGE value '%s'.	Valid range is from 0.0 to 1.0 inclusive.
<b>0x110-02883</b>	USER	system-moab	WARN	MWM_CANNOT_DESTROY_STATIC_RM	Trigger cannot destroy static RM.	A trigger cannot destroy a static resource manager. Refer to trigger 'destroy'.
<b>0x110-02966</b>	USER	system-moab	WARN	MWM_RM_JOB_SUBMIT_FAILURE	RM %s job submit failed: %s.	Error while submitting the job to the resource manager.
<b>0x110-02a0c</b>	USER	system-moab	WARN	MWM_CANNOT_MODIFY_RM_JOB	Cannot modify %s for RM job %s - '%s'.	The listed attribute of the job could not be changed.
<b>0x110-04004</b>	POWER_USER	system-moab	WARN	MWM_TESTING_WARNING	Testing with argument1: %s. and argument2: %s and argument3: %s	Internal error for testing diagnostics.
<b>0x110-0a713</b>	ADMIN	system-moab	WARN	MWM_CANNOT_LOAD_FILE	Cannot load %s file %s.	Failed to load a file into Moab. Make sure it exists and that permissions are correct.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a71a</b>	ADMIN	system-moab	WARN	MWM_FAILED_TO_WAIT_FOR_CHILD	Failed to wait for child, pid: %s, errno: %s (%s).	The wait() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a71d</b>	ADMIN	system-moab	WARN	MWM_CANNOT_CHMOD_FILE	Failure changing permissions of file: '%s' to mode:'%s', errno: %s (%s).	The chmod() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a71e</b>	ADMIN	system-moab	WARN	MWM_CANNOT_OPEN_FILE_WARNING	Cannot open %s file '%s', errno: %s (%s).	The fopen() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a722</b>	ADMIN	system-moab	WARN	MWM_CANNOT_WRITE_FILE_WARNING	Cannot write to file '%s', errno: %s (%s).	The fwrite() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a723</b>	ADMIN	system-moab	WARN	MWM_CANNOT_CLOSE_FILE_DESCRIPTOR	Cannot close file descriptor %s, errno: %s (%s).	The close() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a724</b>	ADMIN	system-moab	WARN	MWM_CANNOT_RENAME_FILE	Failure renaming file '%s' to '%s', errno: %s (%s).	The rename() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a726</b>	ADMIN	system-moab	WARN	MWM_CANNOT_BIND_TO_PORT	Cannot bind to port %s, errno: %s (%s).	The bind() system call failed. Use the errno and associated message to determine possible causes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a72a</b>	ADMIN	system-moab	WARN	MWM_CANNOT_SEND_TO_SOCKET	Cannot send %s byte packet, errno: %s (%s).	The send() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a72c</b>	ADMIN	system-moab	WARN	MWM_FAILED_GETSOCKOPT_WARNING	Cannot get socket %s option, errno: %s (%s).	The getsockopt() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a72d</b>	ADMIN	system-moab	WARN	MWM_FAILED_SETSOCKOPT_WARNING	Cannot set socket %s option, errno: %s (%s).	The setsockopt() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a738</b>	ADMIN	system-moab	WARN	MWM_CANNOT_SET_UMASK	Failure setting umask on file '%s', errno: %s (%s).	The umask() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a73a</b>	ADMIN	system-moab	WARN	MWM_FAILED_FCNTL_WARNING	Cannot set %s option on file descriptor, errno: %s (%s).	The fcntl() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a73b</b>	ADMIN	system-moab	WARN	MWM_CANNOT_STAT_FILE_WARNING	Cannot get stats on file '%s', errno: %s (%s).	The stat() system call failed. Use the errno and associated message to determine possible causes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a73e</b>	ADMIN	system.-moab	WARN	MWM_GET_HOSTNAME_CLIENT	Cannot get hostname of the client, errno: %s (%s).	The getnameinfo() system call failed. Use the errno and associated message to determine possible causes.
<b>0x110-0a74d</b>	ADMIN	system.-moab	WARN	MWM_CONFIG_FILE_NOT_FOUND_WARNING	Cannot locate configuration file '%s' in '%s'.	Check for the existence of this file.
<b>0x110-0a753</b>	ADMIN	system.-moab	WARN	MWM_RM_CONFIG_INVALID_VALUE	Invalid %s value '%s' specified for RM '%s'.	Check the line in the configuration file for the parameter.
<b>0x110-0a754</b>	ADMIN	system.-moab	WARN	MWM_RM_CONFIG_PROCESS_ATTR	Failed to process attribute '%s' for resource manager '%s'.	Check the line in the configuration file for the parameter.
<b>0x110-0a755</b>	ADMIN	system.-moab	WARN	MWM_RM_CONFIG_ATTR	RM attribute '%s' not handled.	Check the line in the configuration file for the parameter.
<b>0x110-0a756</b>	ADMIN	system.-moab	WARN	MWM_RM_CONFIG_TIMEOUT	Resource manager '%s' has a timeout of less than 50 ms.	Check the line in the configuration file for the parameter.
<b>0x110-0a758</b>	ADMIN	system.-moab	WARN	MWM_CONFIG_PARAM_DEFAULT_VALUE	Configuration parameter '%s[%s]' was not assigned a value. Using default.	Check the line in the configuration file to see if this behaviour is desired.
<b>0x110-0a759</b>	ADMIN	system.-moab	WARN	MWM_CONFIG_PARAM_INTEGER_DEFAULT_VALUE	Configuration parameter '%s[%s]' has a value '%s' that is not an integer. Using default.	Check the line in the configuration file for the integer value.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a75a</b>	ADMIN	system-moab	WARN	MWM_CONFIG_PARAM_DOUBLE_DEFAULT_VALUE	Configuration parameter '%s[%s]' has a value '%s' that is not a double. Using default.	Check the line in the configuration file for the double value.
<b>0x110-0a75b</b>	ADMIN	system-moab	WARN	MWM_CONFIG_PARAM_NULL_VALUE	Configuration parameter '%s[%s]' has a NULL value.	Check the line in the configuration file.
<b>0x110-0a75c</b>	ADMIN	system-moab	WARN	MWM_CONFIG_PARAM_INVALID	Configuration parameter '%s' has an invalid value.	Check the line in the configuration file for the attribute.
<b>0x110-0a75e</b>	ADMIN	system-moab	WARN	MWM_CONFIG_PARAM_UNKNOWN	Configuration parameter '%s[%s]' is not defined.	Check the line in the configuration file for the undefined parameter.
<b>0x110-0a75f</b>	ADMIN	system-moab	WARN	MWM_CONFIG_ATTR_EXTRACTION	Configuration parameter '%s[%s]' attribute value '%s' cannot be extracted.	Check the line in the configuration file for the attribute.
<b>0x110-0a760</b>	ADMIN	system-moab	WARN	MWM_INVALID_CONFIG_LINE	Cannot process line '%s'.	Check the line syntax against the documentation.
<b>0x110-0a763</b>	ADMIN	system-moab	WARN	MWM_UNKNOWN_ADMINCFG_PARAMETER	Unknown ADMINCFG parameter '%s'.	Check the syntax in the configuration file.
<b>0x110-0a764</b>	ADMIN	system-moab	WARN	MWM_UNKNOWN_MID_ATTR	Unknown identity attribute '%s'.	Check the MIDCFG lines in the configuration file.
<b>0x110-0a765</b>	ADMIN	system-moab	WARN	MWM_UNKNOWN_AM_ATTR	Unknown account manager attribute '%s'.	Check the AMCFG lines in the configuration file.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a766</b>	ADMIN	system-moab	WARN	MWM_UNKNOWN_ATTRIBUTE_SPECIFIED	Unknown attribute '%s' specified for %s %s.	An error occurred while parsing the configuration for the listed object. The specified attribute is unknown or invalid.
<b>0x110-0a769</b>	ADMIN	system-moab	WARN	MWM_MONGOSERVER_CONNECTION_FAILURE	Unable to connect to Mongo server '%s' (%s).	The program will continue to try and connect in the background.
<b>0x110-0a76e</b>	ADMIN	system-moab	WARN	MWM_EVENT_QUERY_ODBC	Event querying is only supported with ODBC.	Check the USEDATABASE option.
<b>0x110-0a770</b>	ADMIN	system-moab	WARN	MWM_DB_CONNECT	Cannot connect to DB--falling back to file and memory-based storage (%s).	Verify that the database is running.
<b>0x110-0a772</b>	ADMIN	system-moab	WARN	MWM_DATABASE_STATS	Unable to retrieve statistics from the database.	Verify that the database is running.
<b>0x110-0a774</b>	ADMIN	system-moab	WARN	MWM_SERVER_CONNECTION_FAILED_TRYING_FALLBACK	The system was unable to connect to the server %s:%s - attempting fallback server %s.	Make sure the server's address is correct and it is running.
<b>0x110-0a775</b>	ADMIN	system-moab	WARN	MWM_PRIMARY_SERVER_FAILED_TRYING_BACKUP	The system was unable to connect to the server %s (%s:%s) - trying backup server (%s:%s).	Make sure the server's address is correct and it is running.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a778</b>	ADMIN	system-moab	WARN	MWM_INVALID_REDUCE_CLIENTMAXCONNECTIONS	Reducing CLIENTMAXCONNECTIONS to %s from %s not allowed during runtime.	Decreasing the value of CLIENTMAXCONNECTIONS cannot be done during runtime.
<b>0x110-0a785</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_START_JOB	Cannot start job %s. (%s)	The job failed to start.
<b>0x110-0a788</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_ALLOCATE_NODES_FORJOB	Cannot allocate nodes for job %s. (%s)	Cannot allocate a node list that matches the requirements for this job. This may not be serious since multiple passes may occur.
<b>0x110-0a7d2</b>	ADMIN	system-moab	WARN	MWM_UNABLE_LOAD_PBS_JOB	Cannot load PBS job '%s'.	Could not load a job discovered from a PBS resource manager into Moab.
<b>0x110-0a7e3</b>	ADMIN	system-moab	WARN	MWM_TRUNCATING_ATTRIBUTE_FOR_CLASS	Truncating %s for class: %s (rm reports: %s; Moab enforces: %s).	The resource manager reports a certain value for a class, but Moab has been instructed to keep it within certain limits. The value will be truncated to keep it within the limits.
<b>0x110-0a7e4</b>	ADMIN	system-moab	WARN	MWM_UNEXPECTED_JOB_STATE	Unexpected job state '%s' detected for job %s.	The listed job was found to be in a state that was not expected. This may or may not be an error condition.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a821</b>	ADMIN	system.-moab	WARN	MWM_CLOCK_SKEW_DETECTED	Clock skew detected (%s time for job %s in %s).	A reported time associated with the job appears to be wrong. This could be because of a lack of synchronization between system clocks on all nodes.
<b>0x110-0a83e</b>	ADMIN	system.-moab	WARN	MWM_INVALID_WIKI_ATTRIBUTE	Encountered invalid wiki attribute while reading '%s'.	Check the syntax of the attribute.
<b>0x110-0a83f</b>	ADMIN	system.-moab	WARN	MWM_DUPLICATE_WIKI_ATTRIBUTE	Wiki attribute '%s' is already set.	Check for duplicate instances of the attribute.
<b>0x110-0a840</b>	ADMIN	system.-moab	WARN	MWM_VM_UNSUPPORTED_WIKI_ATTRIBUTE	Wiki attribute '%s' is unsupported for VM creation.	Remove the attribute.
<b>0x110-0a843</b>	ADMIN	system.-moab	WARN	MWM_ADD_NODE_FAILURE	Cannot add node '%s' to global node table. Index is already used.	Cannot have two nodes with the same name.
<b>0x110-0a844</b>	ADMIN	system.-moab	WARN	MWM_HT_ADD_NODE_FAILURE	Cannot add node '%s' to hash table. Index is already used.	Cannot have two nodes with the same name.
<b>0x110-0a846</b>	ADMIN	system.-moab	WARN	MWM_VM_MIGRATION_FAILURE	Cannot migrate VM '%s'.	The VM might not be eligible for migration.
<b>0x110-0a84c</b>	ADMIN	system.-moab	WARN	MWM_UNEXPECTED_SUBCOMMAND_RECEIVED	Unexpected subcommand '%s' received.	The communication from a Moab client includes an unknown subcommand.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a84d</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_REGISTER_JOB_AM	Unable to register job %s with accounting manager for job %s. reason: '%s' message: '%s'.	The accounting manager was unable to register the listed job for a certain action. An optional reason and/or message may be given to assist in diagnosis.
<b>0x110-0a851</b>	ADMIN	system-moab	WARN	MWM_IMPROPER_VM_MIGRATION_DECISION	The migration decision for the VM was not properly set up.	The information indicating the destination node is missing.
<b>0x110-0a85b</b>	ADMIN	system-moab	WARN	MWM_JOB_UPDATE_NULL_STARTTIME	Start time is NULL for job update.	Specify a start time that is greater than zero.
<b>0x110-0a85c</b>	ADMIN	system-moab	WARN	MWM_JOB_UPDATE_NULL_DISPATCHTIME	Dispatch time is NULL for job update.	Specify a dispatch time that is greater than zero.
<b>0x110-0a862</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_REGISTER_RESERVATION_AM	Unable to register reservation %s with accounting manager for %s processors for reservation %s.	The accounting manager was unable to register the listed reservation for a certain action.
<b>0x110-0a863</b>	ADMIN	system-moab	WARN	MWM_DEPRECATED_PARAMETER_VALUE	Deprecated value '%s' specified for parameter '%s'. %s	The listed value is no longer valid for this parameter. A hint may be provided with the message. Check the most recent documentation for the software version.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a864</b>	ADMIN	system.-moab	WARNING	MWM_INVALID_FILE_ATTRIBUTES_WARNING	Invalid value '%s' specified for %s (%s).	Checking a file to see whether it exists, is executable, etc, has produced unexpected results.
<b>0x110-0a865</b>	ADMIN	system.-moab	WARNING	MWM_CANNOT_SET_JOB_ATTRIBUTE_VIA_TEMPLATE_WARNING	Cannot set %s %s via template %s.	Failed to set the listed attribute to the listed value for a specified job template.
<b>0x110-0a866</b>	ADMIN	system.-moab	WARNING	MWM_KILL_PROCESS_FAILURE	Unable to kill process %s.	The system tried to kill the given process and failed.
<b>0x110-0a86a</b>	ADMIN	system.-moab	WARNING	MWM_NODELIST_STRING_BUFFER	Insufficient buffer space to convert a node list into a string.	The buffer must be larger to hold all the nodes.
<b>0x110-0a86b</b>	ADMIN	system.-moab	WARNING	MWM_MAX_NODES_EXCEEDED	The maximum number of nodes associated with a reservation has been exceeded.	The number of nodes must be reduced.
<b>0x110-0a871</b>	ADMIN	system.-moab	WARNING	MWM_VC_ALREADY_ADDED	The virtual container '%s' is already an ancestor of VC '%s'.	Cannot create a circular chain, must maintain a hierarchical structure.
<b>0x110-0a872</b>	ADMIN	system.-moab	WARNING	MWM_VC_REMOVAL_FAILURE	The virtual container '%s' cannot be removed.	This is an internal error.
<b>0x110-0a874</b>	ADMIN	system.-moab	WARNING	MWM_RESERVATION_JOB_NOT_FOUND	Unable to find the job for reservation '%s'.	The host job for the reservation is NULL.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a875</b>	ADMIN	system-moab	WARN	MWM_SINGLE_USE_RESERVATION_DESTRUCTION	Unable to destroy a single-use reservation.	This is an internal error.
<b>0x110-0a878</b>	ADMIN	system-moab	WARN	MWM_UNEXPECTED_JOB_SUBMISSION_POLICY	The system encountered an unexpected job submission policy (%s).	The job submission policy did not match a defined policy.
<b>0x110-0a879</b>	ADMIN	system-moab	WARN	MWM_SIMULATION_JOB_RECORDS	Unable to simulate workload by creating job records (1000 attempts).	The system may be low on memory.
<b>0x110-0a880</b>	ADMIN	system-moab	WARN	MWM_JOB_TRANSITION_XML_MESSAGE	Unable to add messages to job '%s' transition XML.	The system may be low on memory.
<b>0x110-0a884</b>	ADMIN	system-moab	WARN	MWM_PBS_API_STALE	PBS API is stale - re-initializing.	Re-initializing the PBS environment.
<b>0x110-0a885</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_GET_PBS_QUEUE_INFO	Cannot process PBS queue info for RM %s (node %s) - no data available.	Unable to get any information on the PBS queues. Make sure that there was at least a queue set up in PBS.
<b>0x110-0a886</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_SET_JOB_ATTRIBUTE	Cannot set job '%s' attribute '%s:%s' to '%s' (rc: %s; '%s').	There was a problem while changing the job attribute and the error status was displayed in rc.
<b>0x110-0a887</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_CONNECT_PBS_SCHEDULER	Cannot connect to PBS event/scheduler port %s.	Ensure the PBS scheduler is running and listening on the specified port.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a888</b>	ADMIN	system.-moab	WARN	MWM_NODE_UNUSABLE_NO_DISK	Idle node %s is unusable (inadequate disk space in /var).	Ensure that the node has sufficient disk space.
<b>0x110-0a889</b>	ADMIN	system.-moab	WARN	MWM_NODE_UNUSABLE_BAD_STATE	Node '%s' is unusable in state 'NONE'.	The node has become unusable because of its state being NONE.
<b>0x110-0a88a</b>	ADMIN	system.-moab	WARN	MWM_UNABLE_TO_FIND_USERS_GID	Cannot locate OS GID information for user '%s' - ignoring user.	Moab was unable to find the GID of this user. Make sure that this user has a GID.
<b>0x110-0a88b</b>	ADMIN	system.-moab	WARN	MWM_UNABLE_TO_FIND_USERS_UID	Cannot locate OS information for user '%s' - ignoring user - %s.	Moab was unable to find the user on the system. Make sure that this user exists.
<b>0x110-0a88c</b>	ADMIN	system.-moab	WARN	MWM_UNABLE_TO_FIND_GID_LIST	Cannot locate OS group list information for user '%s' - ignoring user.	Moab was unable to find the group list for this user.
<b>0x110-0a88d</b>	ADMIN	system.-moab	WARN	MWM_TIMEOUT	Command '%s' timed out, or wait failed after %s seconds.	Increasing the TIMEOUT settings in moab.cfg may help.
<b>0x110-0a88e</b>	ADMIN	system.-moab	WARN	MWM_INSUFFICIENT_POLICIES	Insufficient policies specified; hpolicy=%s,spolicy=%s.	Please revise your policies along with their actions.
<b>0x110-0a88f</b>	ADMIN	system.-moab	WARN	MWM_NO_STDOUT	Request succeeded with no stdout but stderr='%s'.	Typically there will also be stdout when there is stderr. Depending on the request this may be the intended result.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a890</b>	ADMIN	system-moab	WARN	MWM_CANNOT_CONNECT_WIKI	Cannot connect to Wiki event port %s.	Failure to connect to the Wiki event port.
<b>0x110-0a891</b>	ADMIN	system-moab	WARN	MWM_MISSING_COLON_STR	Colon delimiter not located in %s wiki string '%s..' in %s.	Check that the string contains the right format.
<b>0x110-0a892</b>	ADMIN	system-moab	WARN	MWM_ADD_DEPENDENCY_FAIL	Failed to add dependencies to job %s's submission.	There was a problem in adding the dependencies to the job.
<b>0x110-0a893</b>	ADMIN	system-moab	WARN	MWM_INVALID_PRIORITY_FUNCTION	Invalid priority function '%s' on job '%s'.	The priority function applied to the job was invalid.
<b>0x110-0a894</b>	ADMIN	system-moab	WARN	MWM_MISSING_JOB_REQ	Invalid job '%s'; no requirements.	The job was invalid because it was missing the requirements.
<b>0x110-0a895</b>	ADMIN	system-moab	WARN	MWM_MISSING_JOB_REQ_AT_INDEX	Invalid job %s; no requirement at index %s.	The job was invalid because an index was missing requirements.
<b>0x110-0a896</b>	ADMIN	system-moab	WARN	MWM_INVALID_WIKI_STR_MISSING_EQUAL	Malformed wiki string '%s' - no '='.	The wiki string was missing an equal sign '='.
<b>0x110-0a897</b>	ADMIN	system-moab	WARN	MWM_INVALID_EMPTY_WIKI_STR	Malformed wiki string '%s' - EOF.	The wiki string was empty.
<b>0x110-0a898</b>	ADMIN	system-moab	WARN	MWM_INCORRECT_STAGE_LOC	stage-data source location is being incorrectly reported via wiki '%s' != '%s'.	The stage data source location was incorrectly reported in wiki.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a899</b>	ADMIN	system-moab	WARN	MWM_INVALID_VM_OBJECT_ID	VM '%s' is not a valid object, ignoring.	The VM does not have a valid object ID.
<b>0x110-0a89a</b>	ADMIN	system-moab	WARN	MWM_INVALID_JSON_CLUSTER	Could not parse JSON cluster query data from MWS RM (%s): %s.	The JSON construct (s) for the cluster may contain some invalid syntax.
<b>0x110-0a89b</b>	ADMIN	system-moab	WARN	MWM_INVALID_JSON_WORKLOAD	Could not parse JSON workload query data from MWS RM (%s): %s.	The JSON construct (s) for the workload may contain some invalid syntax.
<b>0x110-0a89c</b>	ADMIN	system-moab	WARN	MWM_MISSING_REQ_PROPERTIES	JSON cluster query data from MWS RM (%s) does not contain required properties (%s, %s, %s).	The JSON constructs for the cluster query data are missing the required properties.
<b>0x110-0a89d</b>	ADMIN	system-moab	WARN	MWM_INVALID_JSON_CLUSTER_OBJECT	JSON cluster query data from MWS RM (%s) is not a valid object.	Review the JSON construct for the cluster query data to ensure its syntax is correct.
<b>0x110-0a89e</b>	ADMIN	system-moab	WARN	MWM_INVALID_JSON_WORKLOAD_OBJECT	JSON workload query data from MWS RM (%s) is not a valid object.	Review the JSON construct for the workload query data to ensure its syntax is correct.
<b>0x110-0a89f</b>	ADMIN	system-moab	WARN	MWM_EMPTY_RESPONSE	Empty %s response from RM (%s).	The response from the resource manager query was empty.
<b>0x110-0a8a0</b>	ADMIN	system-moab	WARN	MWM_INVALID_NODE_DATA	Nodes data from MWS RM (%s) is not a valid object.	The response from the Moab Web Services resource manager query was empty.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8a1</b>	ADMIN	system-moab	WARN	MWM_CREATE_RESERVATION_FAIL	Cannot create requested reservation (%s).	The request to create the given reservation has failed.
<b>0x110-0a8a2</b>	ADMIN	system-moab	WARN	MWM_LOCATE_RSVPFILE_FAIL	Cannot locate RSVPROFILE '%s'.	Moab failed to find the given RSVPROFILE. Confirm that the file exists.
<b>0x110-0a8a3</b>	ADMIN	system-moab	WARN	MWM_LOCATE_RSV_PARENT_FAIL	Cannot locate parent '%s' for reservation '%s'.	Moab failed to locate the parent of the given reservation.
<b>0x110-0a8a4</b>	ADMIN	system-moab	WARN	MWM_LOCATE_COMMAND_FAIL	Cannot locate command '%s'.	Moab failed to locate the given command. Confirm that the command exists.
<b>0x110-0a8a5</b>	ADMIN	system-moab	WARN	MWM_UNSUPPORTED_SCHED_CMD	Received unexpected sched command '%s'.	Received an unexpected mschedctl command. Confirm that the used option is supported.
<b>0x110-0a8a6</b>	ADMIN	system-moab	WARN	MWM_UNSUPPORTED_EVENT	Unsupported event '%s' from RM '%s'.	The given event is not supported by the given resource manager.
<b>0x110-0a8a7</b>	ADMIN	system-moab	WARN	MWM_VM_MIGRATE_FAIL	VM %s should migrate from node %s but cannot locate valid destination - %s (policy).	Attempt to migrate the given VM from the given node failed. Please check that the destination is valid.
<b>0x110-0a8a8</b>	ADMIN	system-moab	WARN	MWM_JOB_START_FAIL	Start of system job %s failed; no action specified.	Failed to start a job because there was no action specified.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8a9</b>	ADMIN	system.-moab	WARN	MWM_VMTRACKING_JOB_FAIL	VM '%s' reported a system job failure on VMTracking job '%s'.	The given VM reported it failed on the given VMTracking job.
<b>0x110-0a8aa</b>	ADMIN	system.-moab	WARN	MWM_VMTRACKING_EXCEED_WALTIME	VM '%s' exceeded its allocated walltime. VMTracking job '%s' (pointing to job '%s').	The given VM has exceeded its allocated walltime on the associated VM tracking job.
<b>0x110-0a8ac</b>	ADMIN	system.-moab	WARN	MWM_UNKNOWN_POWER_STATE	No RM can report node '%s' power state for system job '%s'.	No resource manager can report the power state for the given nodes on the given job.
<b>0x110-0a8ad</b>	ADMIN	system.-moab	WARN	MWM_ADD_GLOBAL_NODE_FAIL	Cannot add global node '%s'.	Failed to add the given global node.
<b>0x110-0a8ae</b>	ADMIN	system.-moab	WARN	MWM_UNKNOWN_CLIENT	Client ID '%s' is unknown.	Moab failed to recognize the name/ID of the given client.
<b>0x110-0a8af</b>	ADMIN	system.-moab	WARN	MWM_JOB_DEBIT_ACCOUNT	Unable to charge funds for job.	The account manager failed to debit the account for the job.
<b>0x110-0a8b0</b>	ADMIN	system.-moab	WARN	MWM_JOB_RESERVE_ACCOUNT	Unable to reserve funds for job (Reason: %s).	The account manager failed to reserve funds on the account for the job.
<b>0x110-0a8b1</b>	ADMIN	system.-moab	WARN	MWM_CANCEL_LEIN	Unable to cancel lien for instance '%s' (Reason: %s).	The account manager failed to release the lien.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8b2</b>	ADMIN	system-moab	WARN	MWM_RESERVATION_RESERVE_ACCOUNT	Unable to reserve funds for reservation (Reason: %s).	The account manager failed to reserve funds on the account for the reservation.
<b>0x110-0a8b3</b>	ADMIN	system-moab	WARN	MWM_JOB_TASK_DISTRIBUTION	The system cannot distribute the tasks allocated for a job.	Check the tasks specified in the job.
<b>0x110-0a8b4</b>	ADMIN	system-moab	WARN	MWM_JOB_DEFAULT_CLASS	Job cannot run with default class '%s'.	Check the limits set on the class.
<b>0x110-0a8b5</b>	ADMIN	system-moab	WARN	MWM_RM_START_JOB	Cannot start job through a resource manager.	The resource manager may not be set to run the job.
<b>0x110-0a8b6</b>	ADMIN	system-moab	WARN	MWM_JOB_DEPENDENCY_UPDATE	Cannot find job '%s' to update dependency '%s' for job '%s'.	The dependency job for the specified job is missing.
<b>0x110-0a8b7</b>	ADMIN	system-moab	WARN	MWM_EXPIRED_CHECKPOINT	The checkpoint has expired.	Items within the checkpoint may no longer be valid.
<b>0x110-0a8b8</b>	ADMIN	system-moab	WARN	MWM_BAD_CHECKPOINT_LINE	The system encountered an incorrectly formed checkpoint line for key '%s'.	All lines must end with a NEWLINE character.
<b>0x110-0a8b9</b>	ADMIN	system-moab	WARN	MWM_CONVERT_XML_FROM_STRING	XML data cannot be obtained from an XML string ('%s').	There was an error converting from a string that should contain XML into internal XML data structures.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8ba</b>	ADMIN	system.-moab	WAR-N	MWM_CONVERT_XML_TO_STRING	An XML string cannot be constructed from XML data.	There was an error converting from internal XML data structures into an XML string representation.
<b>0x110-0a8bb</b>	ADMIN	system.-moab	WAR-N	MWM_SOCKET_OPERATION	Cannot %s message on sd %s within %s second timeout.	There is a communication error with sockets.
<b>0x110-0a8bc</b>	ADMIN	system.-moab	WAR-N	MWM_COMPLETED_JOB_RECORD	Could not create job record for completed job %s - %s.	The system may be low on memory.
<b>0x110-0a8bd</b>	ADMIN	system.-moab	WAR-N	MWM_CREATE_TEMPLATE_JOB_DEPENDENCY	Could not create template job dependency %s - %s.	The system may be low on memory.
<b>0x110-0a8be</b>	ADMIN	system.-moab	WAR-N	MWM_CANNOT_FIND_SMP_NODE_BY_FEATURE	Could not find SMP node by feature '%s'.	The feature did not match any of the SMP nodes.
<b>0x110-0a8bf</b>	ADMIN	system.-moab	WAR-N	MWM_CHECKPOINT_PROCESS_COMPLETED_JOB	Could not process completed job from checkpoint.	Examine the checkpoint entry for the job.
<b>0x110-0a8c0</b>	ADMIN	system.-moab	WAR-N	MWM_CANNOT_FIND_SMP_NODE_IN_PARTITION	Could not find SMP node in partition '%s'.	The feature did not match any of the SMP nodes in the partition.
<b>0x110-0a8c1</b>	ADMIN	system.-moab	WAR-N	MWM_JOB_REPORTED_BY_RM_NOT_OWNER	Job '%s' is being reported by RM '%s' but is owned by RM '%s'.	The resource manager reporting does not own the job.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8c2</b>	ADMIN	system-moab	WARN	MWM_PEER_RM_UNKNOWN_LANGUAGE	Peer RM '%s' reported unknown language: '%s'.	The language does not match a known format.
<b>0x110-0a8c3</b>	ADMIN	system-moab	WARN	MWM_PEER_RM_UNKNOWN_SUBLANGUAGE	Peer RM '%s' reported unknown sublanguage: '%s'.	The language does not match a known format.
<b>0x110-0a8c4</b>	ADMIN	system-moab	WARN	MWM_REQ_ATTR_TO_STRING	Required attribute '%s' could not be converted to a string.	There is no string conversion routine for that attribute type.
<b>0x110-0a8c5</b>	ADMIN	system-moab	WARN	MWM_JOB_ARCH_VALUE	Job '%s' does not have a valid arch (architecture) value '%s'.	Check the specified value for the architecture.
<b>0x110-0a8c6</b>	ADMIN	system-moab	WARN	MWM_INVALID_HOST_REQ	Job '%s' does not have a valid host requirement '%s'.	Check the specified value for the requirement.
<b>0x110-0a8c7</b>	ADMIN	system-moab	WARN	MWM_JOB_OPYSYS_VALUE	Job '%s' does not have a valid operating system value '%s'.	Check the specified value for the operating system.
<b>0x110-0a8c8</b>	ADMIN	system-moab	WARN	MWM_UNSUPPORTED_REQ	Resource requirement '%s' not supported.	The requirement specified is unsupported.
<b>0x110-0a8c9</b>	ADMIN	system-moab	WARN	MWM_NO_TASKS	Job loaded in active state with no tasks allocated.	Jobs must have at least one task.
<b>0x110-0a8ca</b>	ADMIN	system-moab	WARN	MWM_SINGLE_ITERATION_JOB_COMPLETION	Scheduler cannot handle job completion in a single iteration.	The job must not start and complete while the scheduler is sleeping.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8cb</b>	ADMIN	system.-moab	WARN	MWM_MAX_TASKS_EXCEEDED	The number of tasks associated with a job has exceeded the maximum (%s).	The number of tasks must be reduced or the scheduler must be configured to accept more tasks.
<b>0x110-0a8cc</b>	ADMIN	system.-moab	WARN	MWM_NODE_OUT_OF_RANGE	Job '%s' node index (%s) at task list index (%s) is out of range.	This is an internal limit.
<b>0x110-0a8cd</b>	ADMIN	system.-moab	WARN	MWM_NODE_NULL	Job '%s' node index (%s) at task list index (%s) is NULL.	This is an internal limit.
<b>0x110-0a8ce</b>	ADMIN	system.-moab	WARN	MWM_NODESET_CONSTRAINTS	Nodeset constraints prevent use of task for job '%s':%s at %s.	The specified nodeset cannot run the task.
<b>0x110-0a8cf</b>	ADMIN	system.-moab	WARN	MWM_DEFAULT_WALLTIME	Job assigned default walltime limit (%s).	Unlimited or no walltime limit specified.
<b>0x110-0a8d0</b>	ADMIN	system.-moab	WARN	MWM_PARTITION_ACCESS	Job cannot access requested partitions (%s).	The partition access list disallows the job.
<b>0x110-0a8d1</b>	ADMIN	system.-moab	WARN	MWM_UNABLE_TO_ALLOCATE_TASKS_FOR_JOB	Cannot allocate tasks for job at %s.	The system may be low on memory.
<b>0x110-0a8d2</b>	ADMIN	system.-moab	WARN	MWM_IGNORE_PARTIAL_RANGE	Ignoring partial time range since full range previously located.	The system will use the full range instead.
<b>0x110-0a8d3</b>	ADMIN	system.-moab	WARN	MWM_DESTINATION_RM	Cannot locate a valid destination resource manager for job.	The submitted job could not be sent to a resource manager.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8d4</b>	ADMIN	system-moab	WARN	MWM_JOB_CREDENTIALS	Cannot authenticate the submitted job (Reason: %s).	The user for the job is not a member of a group or account with access.
<b>0x110-0a8d5</b>	ADMIN	system-moab	WARN	MWM_SMPNODE_BY_FEATURE	Could not find SMPNode by feature %s (%s).	None of the nodes has the feature specified.
<b>0x110-0a8d6</b>	ADMIN	system-moab	WARN	MWM_SMPNODE_BY_PARTITION	Could not find SMPNode in partition %s.	The SMPNode specified is not in the given partition.
<b>0x110-0a8d7</b>	ADMIN	system-moab	WARN	MWM_RSV_ATTR_TO_STRING	Reservation '%s' attribute '%s' could not be converted to a string.	There is no string conversion routine for that attribute type.
<b>0x110-0a8d8</b>	ADMIN	system-moab	WARN	MWM_AM_STATUS	Account manager sent failure message - %s.	Check status message.
<b>0x110-0a8d9</b>	ADMIN	system-moab	WARN	MWM_AM_FAILURE	Native accounting manager call '%s' failed using input XML '%s'.	Check XML syntax.
<b>0x110-0a8da</b>	ADMIN	system-moab	WARN	MWM_AM_INSUFFICIENT_FUNDS	Account manager - Insufficient funds '%s'.	Validate that the user has access to account.
<b>0x110-0a8db</b>	ADMIN	system-moab	WARN	MWM_MIGRATE_JOB	Unable to migrate job '%s' to RM '%s' (%s).	Check the error message.
<b>0x110-0a8dc</b>	ADMIN	system-moab	WARN	MWM_RESERVE_PRIORITY_JOB	Unable to reserve priority job.	Check the error message.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8dd</b>	ADMIN	system.-moab	WARN	MWM_SYNC_JOB	Job '%s' not synchronized to start with job '%s'.	The two jobs must start at the same time.
<b>0x110-0a8de</b>	ADMIN	system.-moab	WARN	MWM_SYNC_JOB_QUEUE	Job '%s' could not start. Requeuing any synchronized jobs.	The other jobs should be back on the queue.
<b>0x110-0a8df</b>	ADMIN	system.-moab	WARN	MWM_UNHANDLED_PLUGIN_EXCEPTION	A node allocation plugin '%s' encountered an unhandled exception '%s'.	Consult the documentation for the plugin.
<b>0x110-0a8e0</b>	ADMIN	system.-moab	WARN	MWM_PLUGIN_LOADED_FAILURE	Error loading node allocation plugin '%s' for partition '%s' %s.	A NodeAllocation plugin was not loaded because of an error. Default node allocation will be used.
<b>0x110-0a8e2</b>	ADMIN	system.-moab	WARN	MWM_UNKNOWN_JOB_DEPENDENCY	Unknown job dependency '%s' on job.	The job is trying to use a dependency that is unknown.
<b>0x110-0a8e3</b>	ADMIN	system.-moab	WARN	MWM_UNSUPPORTED_JOB_DEPENDENCY	Unknown job dependency type '%s' on job.	The job is trying to use a dependency type that is unsupported.
<b>0x110-0a8e4</b>	ADMIN	system.-moab	WARN	MWM_MISSING_JOB_DEPENDENCY	Cannot find dependency job. MissingDependencyAction is '%s'.	Check for the existence of the job dependency.
<b>0x110-0a8e5</b>	ADMIN	system.-moab	WARN	MWM_PARTITION_REP_NODE	Corrupt partition representative node.	Check the representative node for the partition.
<b>0x110-0a8e6</b>	ADMIN	system.-moab	WARN	MWM_PARTITION_ATTRIBUTE	Partition attribute '%s' is not configurable.	Consult the documentation to see which attributes can be configured.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8e7</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_LOCATE_NODE	Unable to locate specified nodes for job.	Could not find a node in the job's node list.
<b>0x110-0a8e8</b>	ADMIN	system-moab	WARN	MWM_NODES_MISSING_FROM_FEASIBLE_LIST	Specified node(s) not found in feasible hostlist.	Could not find a node.
<b>0x110-0a8e9</b>	ADMIN	system-moab	WARN	MWM_HOSTLIST_HAS_TOO_FEW_TASKS	A hostlist has too few tasks available for job '%s': '%s' (%s < %s).	More nodes are needed to satisfy the task requirements.
<b>0x110-0a8ea</b>	ADMIN	system-moab	WARN	MWM_TASKS_REMAINING	A hostlist was unable to handle all tasks (%s remain).	More nodes are needed to satisfy the task requirements.
<b>0x110-0a8eb</b>	ADMIN	system-moab	WARN	MWM_NODE_DOWN	Unable to detect node '%s' for '%s' seconds. Marking it down or removing it.	Make sure the node is up and running.
<b>0x110-0a8ec</b>	ADMIN	system-moab	WARN	MWM_NODE_RESET_EMPTY	Unable to reset node. Node list empty.	Must specify a valid node to reset.
<b>0x110-0a8ed</b>	ADMIN	system-moab	WARN	MWM_NODE_RESET_URL	Unable to reset node. NODEPOWERURL not specified.	Must specify a valid URL for the node to reset.
<b>0x110-0a8ee</b>	ADMIN	system-moab	WARN	MWM_AM_DOWN	The account manager is not currently running.	Check the status of the account manager.
<b>0x110-0a8ef</b>	ADMIN	system-moab	WARN	MWM_KEYBOARD_ACTIVITY_PREEMPT_JOB	Keyboard activity on node prevented job preemption.	Jobs can be preempted only if the keyboard is idle.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8f0</b>	ADMIN	system-moab	WAR-N	MWM_KEYBOARD_ACTIVITY_SET_NODE_STATE	Keyboard activity on node prevented setting the node state to '%s'.	Node states can be changed only if the keyboard is idle.
<b>0x110-0a8f1</b>	ADMIN	system-moab	WAR-N	MWM_JOB_MISMATCHED_TIMES	Fixing job '%s' with invalid '%s' times (%s - %s).	Check the times for the specified job.
<b>0x110-0a8f2</b>	ADMIN	system-moab	WAR-N	MWM_JOB_OPSYS	Cannot add operating system '%s' to job.	Check the type of operating system specified.
<b>0x110-0a8f3</b>	ADMIN	system-moab	WAR-N	MWM_JOB_ARCH	Cannot add architecture '%s' to job.	Check the type of architecture specified.
<b>0x110-0a8f4</b>	ADMIN	system-moab	WAR-N	MWM_LOCATE_AM	Cannot locate the account manager '%s'.	Check the account manager command option syntax.
<b>0x110-0a8f5</b>	ADMIN	system-moab	WAR-N	MWM_LOCATE_RM	Cannot locate the resource manager '%s'.	Check the resource manager command option syntax.
<b>0x110-0a8f6</b>	ADMIN	system-moab	WAR-N	MWM_LOCATE_RMID	Cannot locate the resource manager ID '%s'.	Check the ID command option syntax.
<b>0x110-0a8f7</b>	ADMIN	system-moab	WAR-N	MWM_LOCATE_PARTITION	Cannot locate the partition '%s'.	Check the partition command option syntax.
<b>0x110-0a8f8</b>	ADMIN	system-moab	WAR-N	MWM_RM_QUEUE_MODIFY	Command to modify RM queue failed on resource manager %s - '%s'.	Queue may be configured to reject modify requests.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a8f9</b>	ADMIN	system-moab	WARN	MWM_RM_QUEUE_CREATE	Command to create RM queue failed on resource manager %s - '%s'.	System may be configured to reject queue creation requests.
<b>0x110-0a8fa</b>	ADMIN	system-moab	WARN	MWM_RM_QUEUE_CREATE_MISSING_ARGS	Command to create RM queue failed - arguments missing.	The user must supply the needed arguments to the command.
<b>0x110-0a8fb</b>	ADMIN	system-moab	WARN	MWM_STATIC_RM_DESTRUCTION	An attempt was made to destroy a static resource manager.	Static resource managers cannot be destroyed.
<b>0x110-0a8fc</b>	ADMIN	system-moab	WARN	MWM_ADD_SYSTEM_USER	Unable to create a new user '%s' in the system.	The system may be low on memory.
<b>0x110-0a8fd</b>	ADMIN	system-moab	WARN	MWM_ADD_PARTITION	The system was unable to create partition '%s'.	The system may be low on memory.
<b>0x110-0a8fe</b>	ADMIN	system-moab	WARN	MWM_CORE_LIMIT	System core limit set to %s (complete core files might not be generated).	Expand the system core limit to ensure the complete core dump can be saved.
<b>0x110-0a8ff</b>	ADMIN	system-moab	WARN	MWM_KEY_FILE_PERMISSIONS	The .moab.key file exists, but the file permissions prevent access (%s).	Check the ownership permissions on the file.
<b>0x110-0a900</b>	ADMIN	system-moab	WARN	MWM_STATS_PERIOD_TYPE	The system could not process stats for period type %s.	'Day' is the only period type currently supported.
<b>0x110-0a901</b>	ADMIN	system-moab	WARN	MWM_STATS_BUFFER_SIZE	The system could not process stats for period type %s (buffer too small).	The buffer allocated was too small to hold the data.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a902</b>	ADMIN	system-moab	WARN	MWM_STATS_FILE	The system could not create the stats file '%s'.	Check the path and user permissions on the directory.
<b>0x110-0a903</b>	ADMIN	system-moab	WARN	MWM_JOB_NO_ACCOUNT	No account specified for job '%s'.	Check the job for an account specification.
<b>0x110-0a904</b>	ADMIN	system-moab	WARN	MWM_SET_JOBATTR_FAIL	Cannot set attribute '%s' to value '%s' on jobmatch '%s'.	Failed to set the given attribute to the given value on the given job.
<b>0x110-0a905</b>	ADMIN	system-moab	WARN	MWM_JOBATTR_NOT_SUPPORTED	JobAttr not supported. '%s'.	The given attribute is not a supported job attribute.
<b>0x110-0a906</b>	ADMIN	system-moab	WARN	MWM_INVALID_TRIGGER_DEFINITION	Invalid trigger definition: %s.	The given trigger is invalid. Check that the given trigger has been defined.
<b>0x110-0a907</b>	ADMIN	system-moab	WARN	MWM_ATTRIBUTE_NOT_HANDLED	System attribute '%s' not handled.	Check that the given attribute was spelled correctly.
<b>0x110-0a908</b>	ADMIN	system-moab	WARN	MWM_QOS_IN_PARAM_NOT_FOUND	Cannot locate QOS '%s' for parameter %s.	Make sure that the given QOS exists.
<b>0x110-0a909</b>	ADMIN	system-moab	WARN	MWM_INVALID_PROFILEDURATION_VAL	Invalid PROFILEDURATION specified, modified internally to %s (see documentation).	The entered PROFILEDURATION value is invalid. Moab uses the given value instead.
<b>0x110-0a90a</b>	ADMIN	system-moab	WARN	MWM_NO_DATA_STAGING_PATH	No path in data staging specification '%s' (bad format).	Verify the data staging path is specified.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a90b</b>	ADMIN	system-moab	WARN	MWM_UNSUPPORTED_RM_DATA_STAGING	Cannot stage-out stdout/stderr (unsupported RM type '%s').	Failed to stage-out stdout/stderr because the given resource manager does not support such feature.
<b>0x110-0a90c</b>	ADMIN	system-moab	WARN	MWM_RM_DATA_ON_NON_EXISTING_JOB	Storage RM '%s' reporting data operation for non-existent job '%s'.	The given resource manager is reporting data operation on the non-existing job.
<b>0x110-0a90d</b>	ADMIN	system-moab	WARN	MWM_INVALID_RM_DATA_STAGE	Data stage for RM '%s' not possible as it has no nodelist.	Check CLUSTERQUERYURL to ensure it at least has a nodelist.
<b>0x110-0a90e</b>	ADMIN	system-moab	WARN	MWM_DATA_STAGE_IN_FAIL	Data stage in failed for job '%s' file '%s' (%s).	Failed to complete the data staging in operation for the job on the given file due to error(s).
<b>0x110-0a90f</b>	ADMIN	system-moab	WARN	MWM_UNABLE_TO_REMOVE_DATA_STAGE	Cannot remove data staging block for job '%s'.	Failed to remove the data staging block for the given job.
<b>0x110-0a910</b>	ADMIN	system-moab	WARN	MWM_DATA_STAGE_OUT_FAIL	Data stage out failed for job '%s' file '%s' (%s).	Failed to complete the data staging out operation for the job on the given file due to error(s).
<b>0x110-0a911</b>	ADMIN	system-moab	WARN	MWM_JOB_INVALID_OPSYS	Job '%s' cannot request OS '%s'.	The requested operating system is not available for the job.
<b>0x110-0a912</b>	ADMIN	system-moab	WARN	MWM_NODE_BUFFER_OVERFLOW	Node buffer is full.	Try increasing the node buffer size.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a913</b>	ADMIN	system-moab	WARN	MWM_REMOVE_NODE_WITH_RESERVATION	Unable to remove node '%s' because of reservation references.	Remove the reservations from the node.
<b>0x110-0a914</b>	ADMIN	system-moab	WARN	MWM_JOB_PURGE_RM_INACTIVE	Unable to purge job '%s' because the resource manager '%s' is inactive.	Check the status of the resource manager.
<b>0x110-0a915</b>	ADMIN	system-moab	WARN	MWM_JOB_PURGE_RM_NO_RESOURCES	Unable to purge job '%s' because the resource manager '%s' is reporting no resources.	Check the status of the resource manager.
<b>0x110-0a916</b>	ADMIN	system-moab	WARN	MWM_JOB_NOT_DETECTED	Job '%s' in state '%s' no longer detected (Last Detected %s > PurgeTime %s).	The job may have been purged in the meantime.
<b>0x110-0a917</b>	ADMIN	system-moab	WARN	MWM_BACKFILL_DEPTH_REACHED	The backfill depth (BFDEPTH) has been reached so no more jobs will be back-filled this iteration.	Wait for the next iteration or increase the depth.
<b>0x110-0a918</b>	ADMIN	system-moab	WARN	MWM_JOB_NO_QUEUE_TIME	No QueueTime has been specified for job.	Configure the job with a queue time.
<b>0x110-0a91c</b>	ADMIN	system-moab	WARN	MWM_CHECKPOINT_CREATE_RSV_FROM_XML	Unable to create a reservation from checkpoint XML.	The system may be low on memory.
<b>0x110-0a91d</b>	ADMIN	system-moab	WARN	MWM_JOB_CACHE_REMOVAL	Failed to remove job %s (ID = %s) from the cache.	The job was missing from the system hash table.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a91e</b>	ADMIN	system-moab	WARN	MWM_INVALID_CRED_VALUE	Invalid credential value '%s'.	Check the syntax in the configuration file.
<b>0x110-0a91f</b>	ADMIN	system-moab	WARN	MWM_INVALID_CRED_ATTR	Invalid credential attribute '%s'.	Check the syntax in the configuration file.
<b>0x110-0a920</b>	ADMIN	system-moab	WARN	MWM_MAX_JOBS_EXCEEDED	The maximum number of jobs has been exceeded.	Increase the value of the MAXJOB setting.
<b>0x110-0a921</b>	ADMIN	system-moab	WARN	MWM_JOB_FAILED_PROCESSING_PBS	A job failed while processing PBS resources.	May not have been able to locate host or vnode.
<b>0x110-0a922</b>	ADMIN	system-moab	WARN	MWM_FIND_PEER	Cannot find client peer for job %s (Name: %s).	The resource manager cannot be located.
<b>0x110-0a923</b>	ADMIN	system-moab	WARN	MWM_AM_INSUFFICIENT_BALANCE	Insufficient balance in primary account '%s' to run job '%s' (attempting fallback credentials).	Validate that the user has access to account.
<b>0x110-0a924</b>	ADMIN	system-moab	WARN	MWM_AM_JOB_SUBMIT_VALIDATION	Job submission validation failed for job '%s' -- taking action '%s'.	Validate that the job has access.
<b>0x110-0a925</b>	ADMIN	system-moab	WARN	MWM_TOO_MANY_NODE_SETS	The maximum number of node sets has been exceeded.	This is a configurable setting.
<b>0x110-0a926</b>	ADMIN	system-moab	WARN	MWM_CLASS_SET_LIST_INVALID	The specified class set list is invalid '%s'.	Check the documentation for valid classes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a928</b>	ADMIN	system.-moab	WAR-N	MWM_RM_START	Cannot start resource manager '%s' (Reason: %s).	The resource manager may not be available.
<b>0x110-0a929</b>	ADMIN	system.-moab	WAR-N	MWM_STDOUT_FAIL	Request succeeded with no stdout. stderr= '%s'.	The standard out may not have been specified.
<b>0x110-0a92a</b>	ADMIN	system.-moab	WAR-N	MWM_UNSUPPORTED_REQ_ATTR	Unsupported req attribute '%s'.	The attribute is not one that can be set.
<b>0x110-0a92b</b>	ADMIN	system.-moab	WAR-N	MWM_UNSUPPORTED_GENERAL_ATTR	Unsupported general attribute '%s'.	The attribute is not one that can be set.
<b>0x110-0a92e</b>	ADMIN	system.-moab	WAR-N	MWM_CREATE_ACCOUNT	Unable to create account '%s' on the account manager.	Verify that the account manager is running.
<b>0x110-0a92f</b>	ADMIN	system.-moab	WAR-N	MWM_QUERY_ACCOUNT	Unable to query account '%s' on the account manager.	Verify that the account name is correct.
<b>0x110-0a930</b>	ADMIN	system.-moab	WAR-N	MWM_ACCOUNT_ADD_USER	Unable to add user '%s' to account '%s' on the account manager.	Verify that the account name is correct.
<b>0x110-0a931</b>	ADMIN	system.-moab	WAR-N	MWM_ACCOUNT_DEPOSIT	Unable to deposit '%s' credits to account '%s' on the account manager.	Verify that the account name is correct.
<b>0x110-0a932</b>	ADMIN	system.-moab	WAR-N	MWM_ALLOCATE_REQ	Unable to allocate requirement '%s' using NAllocPolicy '%s' (%s).	The system may be low on memory.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a933</b>	ADMIN	system-moab	WARN	MWM_VALID_STAT_DATA	Unable to generate valid statistic data for external query.	The system may be low on memory.
<b>0x110-0a934</b>	ADMIN	system-moab	WARN	MWM_JOB_QOS_REQUEST	Job '%s' cannot request QOS '%s').	The requested QOS is not available for the job.
<b>0x110-0a938</b>	ADMIN	system-moab	WARN	MWM_GRES_OVERFLOW	GRES overflow.	Unable to add another GRES.
<b>0x110-0a939</b>	ADMIN	system-moab	WARN	MWM_RM_NO_RESOURCES	The resource manager '%s' is reporting no resources.	Check the nodes on the resource manager.
<b>0x110-0a93d</b>	ADMIN	system-moab	WARN	MWM_EMPTY_FILE	File '%s' is empty.	Check the file specified.
<b>0x110-0a93f</b>	ADMIN	system-moab	WARN	MWM_PREEMPT_NONACTIVE_JOB	Cannot preempt non-active job '%s' (state: '%s' estate: '%s').	The job must currently be active to preempt it.
<b>0x110-0a940</b>	ADMIN	system-moab	WARN	MWM_QUEUE_NONSTARTABLE_JOB	Cannot requeue non-startable job '%s' (canceling instead).	The job could not be requeued.
<b>0x110-0a941</b>	ADMIN	system-moab	WARN	MWM_NODE_RACK_VALUE	Invalid rack value '%s' specified for node %s (must be digit).	Check the value of the rack parameter.
<b>0x110-0a942</b>	ADMIN	system-moab	WARN	MWM_ENCODE_JOB_MESSAGE	Cannot encode job message.	Check the value of the rack parameter.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a943</b>	ADMIN	system.-moab	WARN	MWM_INVALID_PROFILECOUNT_VAL	Invalid PROFILECOUNT specified, modified internally to %s (see documentation).	The PROFILECOUNT value is invalid. Moab uses the default value instead.
<b>0x110-0a944</b>	ADMIN	system.-moab	WARN	MWM_FAIRSHARE_FILE	Cannot load fairshare file '%s' for slot %s.	Check for the existence of the fairshare file in the file system.
<b>0x110-0a945</b>	ADMIN	system.-moab	WARN	MWM_REQ_ATTR_ALREADY_SET	Requirement attribute %s '%s' is already set.	Check the attribute setting in the configuration file.
<b>0x110-0a946</b>	ADMIN	system.-moab	WARN	MWM_ZERO_START_TIME	StartTime set to zero for reservation on job '%s'.	Check the start time for the specified job.
<b>0x110-0a947</b>	ADMIN	system.-moab	WARN	MWM_EXISTING_RESERVATION	Reservation created for reserved job '%s' (existing reservation '%s' deleted).	Only one reservation can exist at a time for the reserved job.
<b>0x110-0a948</b>	ADMIN	system.-moab	WARN	MWM_CANNOT_PARSE_REQ_LINE	Cannot parse requirement line for job '%s'.	The syntax of the requirement line is incorrect.
<b>0x110-0a949</b>	ADMIN	system.-moab	WARN	MWM_UNKNOWN_RESOURCE_TYPE	Unknown resource type '%s' for job '%s'.	Check the documentation for valid resource types.
<b>0x110-0a94a</b>	ADMIN	system.-moab	WARN	MWM_UNKNOWN_TRANSACTION_ATTR	Unknown transaction attribute '%s'.	Check the documentation for valid transaction attributes.
<b>0x110-0a94c</b>	ADMIN	system.-moab	WARN	MWM_ID_MANAGER_DOWN	The identity manager is down.	Check the status of the identity manager.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a94d</b>	ADMIN	system-moab	WARN	MWM_PROCESS_ID_LINE	Unable to process the ID line '%s'.	Check the syntax of the attribute/value pairs.
<b>0x110-0a94e</b>	ADMIN	system-moab	WARN	MWM_JOB_ID_MISSING	Unable to locate the job ID for a job submitted to the resource manager.	Check the job being submitted`.
<b>0x110-0a950</b>	ADMIN	system-moab	WARN	MWM_EMPTY_NODELIST	The nodelist is empty for reservation '%s'.	Reservations should include a node list.
<b>0x110-0a951</b>	ADMIN	system-moab	WARN	MWM_FAIRSHARE_PAL	Fairshare does not allow specified PAL (%s).	The fairshare algorithm is reverting to the original PAL.
<b>0x110-0a952</b>	ADMIN	system-moab	WARN	MWM_JOB_START_TIME	Cannot find earliest start time for job '%s'.	Resources needed to run the job may never be available.
<b>0x110-0a953</b>	ADMIN	system-moab	WARN	MWM_NODE_MODIFY	Cannot modify node '%s' Error(%s).	The node could not be modified.
<b>0x110-0a954</b>	ADMIN	system-moab	WARN	MWM_TRIGGER_RSV_CREATE	Unable to create a trigger reservation.	Check the reservation time, nodes, and account.
<b>0x110-0a955</b>	ADMIN	system-moab	WARN	MWM_MISSING_TRIGGER	Trigger '%s' with PID '%s' does not exist--completing!	The process may have already completed.
<b>0x110-0a956</b>	ADMIN	system-moab	WARN	MWM_EMPTY_HOSTLIST	The hostlist is empty for reservation.	Reservations should include a hostlist.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a957</b>	ADMIN	system-moab	WARN	MWM_RSV_POLICY_VIOLATION	Unable to create requested reservation due to a policy violation (%s).	Reservations must conform to existing policies.
<b>0x110-0a958</b>	ADMIN	system-moab	WARN	MWM_RSV_CREATE_FAILURE	Unable to create requested reservation at time %s (%s).	Resources are unavailable at requested time.
<b>0x110-0a959</b>	ADMIN	system-moab	WARN	MWM_RSV_OWNER	Cannot process owner '%s' for standing reservation '%s' (%s).	Consult the error message.
<b>0x110-0a95a</b>	ADMIN	system-moab	WARN	MWM_RSV_PARTIAL	Partial reservation %s reserved %s of %s procs in partition '%s' to start in %s at (%s) %s.	Entire reservation could not be filled.
<b>0x110-0a95c</b>	ADMIN	system-moab	WARN	MWM_RSV_NEGATIVE_JOBCOUNT	Reservation %s job-count is %s, should not decrement less than 0.	JobCount cannot be negative.
<b>0x110-0a95d</b>	ADMIN	system-moab	WARN	MWM_EMPTY_REQ_NODELIST	Req node list empty for job %s:%s in state %s (job nodelist copied to req nodelist).	Job should include a req node list.
<b>0x110-0a95e</b>	ADMIN	system-moab	WARN	MWM_TASK_ALLOCATION_INFO	Cannot locate task allocation info for job %s:%s in state %s.	Job should include a task list.
<b>0x110-0a95f</b>	ADMIN	system-moab	WARN	MWM_INVALID_THREADPOOL_SIZE	Invalid ThreadPoolSize '%s' (must be a non-negative integer no larger than %s).	Check the size for a valid value.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a960</b>	ADMIN	system-moab	WARN	MWM_INVALID_QUEUE_TIME	Job '%s' has invalid system queue time (SQ: %s > ST: %s).	Check the job queue time value.
<b>0x110-0a961</b>	ADMIN	system-moab	WARN	MWM_NO_WCLIMIT	Job '%s' has no WCLimit specified.	Check the job for the correct value.
<b>0x110-0a962</b>	ADMIN	system-moab	WARN	MWM_AM_INVALID_PROTOCOL	Invalid protocol '%s' specified for account manager '%s'.	Communication with the account manager must be over a supported protocol.
<b>0x110-0a963</b>	ADMIN	system-moab	WARN	MWM_NO_POWER_INTERFACE	No external power interface - cannot set power state '%s' on node '%s%s%s'.	Cannot set the power state on the node without a power interface.
<b>0x110-0a964</b>	ADMIN	system-moab	WARN	MWM_JOB_STARTED_ON_ANOTHER_RM	Job '%s' started externally: (rc: %s; errmsg: '%s'; Tasklist: '%s').	Two or more resource managers are running side-by-side and the job is already running on one of them.
<b>0x110-0a965</b>	ADMIN	system-moab	WARN	MWM_COMMAND_FAILED_CHILD_PROCESS	Job submit request failed with child process status code=%s', stderr='%s', stdout='%s', EMSg='%s'.	Review the status code and error message for further information.
<b>0x110-0a967</b>	ADMIN	system-moab	WARN	MWM_AM_REGISTER_JOB	Unable to register job creation with account manager for job '%s', reason: '%s'.	Check the status of the account manager.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0a968</b>	ADMIN	system.-moab	WARN	MWM_AM_DEPRECATED_PARAMETER	Use of the JOBFAILUREACTION parameter is deprecated. Use STARTFAILUREACTION instead.	Check the documentation for the new parameter syntax.
<b>0x110-0a969</b>	ADMIN	system.-moab	WARN	MWM_AM_INVALID_ACTION	Invalid action '%s' specified in '%s' for account manager '%s'.	Check the documentation for valid actions for the account manager.
<b>0x110-0a9bf</b>	ADMIN	system.-moab	WARN	MWM_ERASING_JOB	Erasing job '%s' by address.	The specified job could not be found by name. The entire job table was searched to find the matching job.
<b>0x110-0a9e1</b>	ADMIN	system.-moab	WARN	MWM_JOB_INVALID_TASK_LAYOUT	Job '%s' has invalid task layout (TPN:%s * N:%s != T:%s).	The task layout does not compute.
<b>0x110-0a9e2</b>	ADMIN	system.-moab	WARN	MWM_JOB_ACCESS_QOS	Job '%s' does not have access to QOS '%s' (QAL: %s).	The QoS is not accessible from the job.
<b>0x110-0a9f8</b>	ADMIN	system.-moab	WARN	MWM_DUPLICATE_SYSTEMJID	Duplicate SystemJID '%s' [JState: %s] found from RM '%s'.	The SystemJID must be unique.
<b>0x110-0a9fc</b>	ADMIN	system.-moab	WARN	MWM_CANNOT_PING_RM	Cannot ping RM '%s' because a file was not specified.	A file path to a valid file is needed.
<b>0x110-0aa07</b>	ADMIN	system.-moab	WARN	MWM_VM_CONTAINER_NODE	Cannot find or add container node '%s' for VM '%s'.	The node could not be found.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0e710</b>	INTERNAL	system-moab	WARN	MWM_INVALID_ARG_VALUE	Invalid arguments passed to this function.	One or more arguments passed to this function were not valid. This is an internal error logged for informational purposes.
<b>0x110-0e72b</b>	INTERNAL	system-moab	WARN	MWM_SEND_SENT_NO_DATA	No data was sent to the socket when it should have been.	The send() system call reported no data was sent when data should have been sent.
<b>0x110-0e72e</b>	INTERNAL	system-moab	WARN	MWM_SOCKET_BLOCKED_UNEXPECTEDLY	Read operations on the socket were blocked when it should have been available.	A socket operation reported that the operation was blocked. Previous information indicated that this operation should have been available.
<b>0x110-0e73f</b>	INTERNAL	system-moab	WARN	MWM_MUTEX_LOCK	Cannot lock mutex semaphore using pthread_mutex_lock().	This is an operating system call problem.
<b>0x110-0e740</b>	INTERNAL	system-moab	WARN	MWM_MUTEX_UNLOCK	Cannot unlock mutex semaphore using pthread_mutex_unlock().	This is an operating system call problem.
<b>0x110-0e771</b>	INTERNAL	system-moab	WARN	MWM_THREAD_DB_INIT	Thread %s attempting to re-initialize database info struct.	Internal error condition.
<b>0x110-0e84b</b>	INTERNAL	system-moab	WARN	MWM_CORRUPT_COMMAND_RECEIVED	Corrupt command '%s' received.	The communication packet received from a Moab client command is malformed.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0e919</b>	INTERNAL	system.-moab	WAR-N	MWM_CHECKPOINT_NO_XML	The checkpoint data does not contain XML.	Internal error.
<b>0x110-0e91a</b>	INTERNAL	system.-moab	WAR-N	MWM_CHECKPOINT_INVALID_XML	The checkpoint data does not contain valid XML (%s).	Internal error.
<b>0x110-0e91b</b>	INTERNAL	system.-moab	WAR-N	MWM_CHECKPOINT_UPDATE_RSV_FROM_XML	Unable to update a reservation from checkpoint XML.	Internal error.
<b>0x110-0e927</b>	INTERNAL	system.-moab	WAR-N	MWM_JOB_ATTR_TO_STRING	Job attribute '%s' not yet translated to string value.	Internal warning.
<b>0x110-0e92c</b>	INTERNAL	system.-moab	WAR-N	MWM_MISSING_STATUS_CODE	The status code was missing from the S3 response.	This is an internal error.
<b>0x110-0e92d</b>	INTERNAL	system.-moab	WAR-N	MWM_MISSING_STATUS_VALUE	The status value was missing from the S3 response.	This is an internal error.
<b>0x110-0e93a</b>	INTERNAL	system.-moab	WAR-N	MWM_SIMULATION_NO_JOBS	No jobs loaded in simulation.	Internal simulation error.
<b>0x110-0e93b</b>	INTERNAL	system.-moab	WAR-N	MWM_SIMULATION_JOB_DETECTED_TRACEBUFFER	Job '%s' previously detected in tracefile (MJobTraceBuffer [%s]/JC: %s; IT: %s).	Internal simulation error.
<b>0x110-0e93c</b>	INTERNAL	system.-moab	WAR-N	MWM_SIMULATION_JOB_DETECTED	Job '%s' previously detected in tracefile (Job/JC: %s; IT: %s).	Internal simulation error.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x110-0e93e</b>	INTERNAL	system-moab	WARN	MWM_READ_COMMAND_OUTPUT	Cannot read output of command '%s'.	This is an internal communications error.
<b>0x110-0e94b</b>	INTERNAL	system-moab	WARN	MWM_THREAD_TIMEOUT	Thread %s killed (%s micro-second time out reached).	This is an internal issue.
<b>0x110-0e94f</b>	INTERNAL	system-moab	WARN	MWM_INVALID_XML_RM	Invalid XML data for resource manager '%s'.	Check the XML syntax.
<b>0x210-00067</b>	USER	domain.lifecycle	ERROR	MWM_JOB_END_FAILED	Job %s failed. %s	The job finished unsuccessfully.
<b>0x210-000ca</b>	USER	domain.lifecycle	ERROR	MWM_NODE_EVAC_VMS_ERROR	Error evacuating VMs off node %s. %s	There was an error while attempting to evacuate the VMs off the node.
<b>0x210-02882</b>	USER	system-moab	ERROR	MWM_NODE_MODIFY_FAILURE	Cannot modify node state of '%s' Error (%s).	The node state could not be modified.
<b>0x210-02a1a</b>	USER	system-moab	ERROR	MWM_DEPRECATED_RM_FEATURE	RM flag SUBMITJOBSASROOT not supported with this version, %s. Must be >= 2.4.8.	The resource manager version should be updated to get support for this feature.
<b>0x210-08003</b>	ADMIN	system-moab	ERROR	MWM_TESTING_ERROR	Testing with argument1: %s. and argument2: %s.	Internal error for testing diagnostics.
<b>0x210-08263</b>	ADMIN	domain.lifecycle	ERROR	MWM_VC_SCHEDULE_FAILURE	Failed to schedule virtual container '%s'.	This is an internal error.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a718</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_FORK_ERROR	Cannot fork the process, errno: %s (%s).	The fork() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a719</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_EXEC_PROGRAM	Cannot exec action '%s', errno: %s (%s).	The exec() system call failed to execute the command. This may be because the command does not exist or the permissions do not allow it to be run. Use the errno and associated message to determine possible causes.
<b>0x210-0a71b</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_CHOWN_FILE	Failure changing ownership of file: '%s' to uid:'%s', gid:'%s', errno: %s (%s).	The chown() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a71f</b>	ADMIN	system.-moab	ERROR	MWM_PIPE_READ_FAILED	Failed to read pipe on command '%s', errno: %s (%s).	The fread() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a720</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_READ_FILE	Cannot read file '%s', errno: %s (%s).	The fread() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a721</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_WRITE_TO_FILE	Failure writing to file, errno: %s (%s).	The write() system call failed. Use the errno and associated message to determine possible causes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a725</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_GET_HOSTNAME	Cannot get hostname '%s', errno: %s (%s).	The gethostname() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a727</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_SOCKET	Failure creating a socket, errno: %s (%s).	The socket() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a728</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CONNECT_TO_HOST	Failure connecting to server '%s' on port %s, errno: %s (%s).	The connect() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a730</b>	ADMIN	system-moab	ERROR	MWM_EPOCH_FAIL	Epoch Fail, time: '%s' cannot be converted to an epoch time, errno: %s (%s).	The mktime() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a731</b>	ADMIN	system-moab	ERROR	MWM_MEMORY_ALLOCATION_FAILURE_MALLOC	Failure allocating memory (malloc), allocating '%s' bytes, errno: %s (%s).	The malloc() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a732</b>	ADMIN	system-moab	ERROR	MWM_MEMORY_ALLOCATION_FAILURE_CALLOC	Failure allocating memory (calloc), allocating '%s' elements of size '%s' bytes, errno: %s (%s).	The calloc() system call failed. Use the errno and associated message to determine possible causes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a733</b>	ADMIN	system.-moab	ERROR	MWM_MEMORY_ALLOCATION_FAILURE_REALLOC	Failure allocating memory (realloc), allocating '%s' bytes, errno: %s (%s).	The realloc() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a734</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_DUPLICATE_STRING	Failure duplicating string, errno: %s (%s).	The strdup() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a735</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_CHANGE_PROCESS_GROUP	Failure changing process group, errno: %s (%s).	The setpgrp() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a736</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_CREATE_THREAD	Failure creating thread: '%s', errno: %s (%s).	The pthread_create() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a737</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_TRUNCATE_FILE	Failure truncating a file '%s', errno: %s (%s).	The truncate() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a739</b>	ADMIN	system.-moab	ERROR	MWM_PIPE_OPEN_FAILED	Failed to open pipe on command '%s', errno: %s (%s)	The popen() system call failed. Use the errno and associated message to determine possible causes.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a73d</b>	ADMIN	system-moab	ERROR	MWM_CHANGE_DIR_FAILURE	OS call to change directory to '%s' failed errno: %s (%s).	The chdir() system call failed. Use the errno and associated message to determine possible causes.
<b>0x210-0a74a</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_LOCK_MOAB_PID_FILE	Cannot lock the PID file '%s'. Is Moab already running?	Moab tries to ensure that only one instance of itself is running. In the default configuration it will exit if it cannot obtain a lock.
<b>0x210-0a74e</b>	ADMIN	system-moab	ERROR	MWM_CONFIG_FILE_NOT_FOUND_ERROR	Cannot locate configuration file in any predetermined location.	Moab cannot find the configuration file. Verify that it is present and installed in a proper location.
<b>0x210-0a757</b>	ADMIN	system-moab	ERROR	MWM_MWS_RM_CONFIGURATION	The resource manager with Moab Web Services (%s) does not have a base URL, username, and password configured.	Correctly configure the Moab Web Services resource manager.
<b>0x210-0a761</b>	ADMIN	system-moab	ERROR	MWM_STRICT_INVALID_CONFIG_LINE	Error processing line #%s: %s - (%s).	Check the line number in the configuration file.
<b>0x210-0a767</b>	ADMIN	system-moab	ERROR	MWM_NO_MONGOSERVER_SPECIFIED	Failed to initialize connection to Mongo server. (Moab is configured to use Mongo, but no MONGOSERVER is specified.)	Cannot connect to the Mongo server since the MONGOSERVER parameter was unspecified. Add MONGOSERVER parameter to moab.cfg.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a768</b>	ADMIN	system.-moab	ERROR	MWM_MONGOSERVER_INITIALIZATION_FAILED	Failed to initialize connection to Mongo server '%s'.	Failed to initialize connection to the configured MONGOSERVER. Check the following: (1) network connection to Mongo server; and (2) check MONGouser and MONGOPASSWORD parameters in moab-private.cfg.
<b>0x210-0a76a</b>	ADMIN	system.-moab	ERROR	MWM_MONGOSERVER_AUTHENTICATION_FAILURE	Failed to authenticate to Mongo server (%s).	Check user credentials.
<b>0x210-0a76b</b>	ADMIN	system.-moab	ERROR	MWM_MONGOSERVER_WRITE_FAILURE	Unable to write out transition object '%s'.	The BSON information is invalid or missing.
<b>0x210-0a76c</b>	ADMIN	system.-moab	ERROR	MWM_MONGOSERVER_DOWN	The Mongo server is down.	Check the status of the server.
<b>0x210-0a76f</b>	ADMIN	system.-moab	ERROR	MWM_DB_CHECKPOINT_OBJECT	Unable to checkpoint object to the database (%s).	Make sure the database is running.
<b>0x210-0a773</b>	ADMIN	system.-moab	ERROR	MWM_BACKUP_SERVER_CONNECTION_FAILED	The system was unable to connect to the backup server %s (%s:%s).	Make sure the backup server's address is correct.
<b>0x210-0a776</b>	ADMIN	system.-moab	ERROR	MWM_CONNECTION_REFUSED	Connection to the server was refused (%s).	Primary server refused and no fallback server available.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a777</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_TO_CONNECT	Cannot send request to %s:%s (%s may not be running).	Unable to connect to the scheduler program.
<b>0x210-0a779</b>	ADMIN	system-moab	ERROR	MWM_CLIENT_MAX_CONNECTIONS_REACHED	Cannot accept connection number %s (transaction number %s) from '%s' (limit reached).	May need to increase the CLIENTMAXCONNECTIONS configuration setting.
<b>0x210-0a77a</b>	ADMIN	system-moab	ERROR	MWM_SERVER_CONNECTION_FAILED	The system was unable to connect to the server %s:%s - %s.	Make sure the server's address is correct and it is running.
<b>0x210-0a77c</b>	ADMIN	system-moab	ERROR	MWM_COMMUNICATION_ERROR	Communication error %s:%s (%s).	General error trying to communicate with the host.
<b>0x210-0a77d</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_PARSE_SERVER_RESPONSE_STATUS	Cannot parse server response (status).	The response sent from the server is malformed.
<b>0x210-0a77e</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_PARSE_SERVER_RESPONSE_DATA	Cannot parse server response (data).	The response sent from the server is malformed.
<b>0x210-0a77f</b>	ADMIN	system-moab	ERROR	MWM_INVALID_FS_TARGET	Invalid type specified for FSTarget.	Fairshare target type is invalid.
<b>0x210-0a780</b>	ADMIN	system-moab	ERROR	MWM_COULD_NOT_ADD_FS_TREE_NODE	Could not add fstree node %s.	Unable to add a node to the fairshare configuration tree.
<b>0x210-0a781</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_ADD_MANAGER_TO_FS_TREE	Could not add manager %s to fstree.	Unable to add a manager to the fairshare tree.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a782</b>	ADMIN	system-moab	ERROR	MWM_CRED_MANAGER_OVERFLOW	CredManager overflow, manager %s not added.	Credential Manager could not add another manager.
<b>0x210-0a783</b>	ADMIN	system-moab	ERROR	MWM_CRED_MANAGER_OVERFLOW_CHILD	CredManager overflow while adding managers to child in fstree.	Fairshare tree configuration problem.
<b>0x210-0a786</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_TO_SELECT_TASKS_FORJOB	Cannot select tasks for job %s. (%s)	Cannot select a node list that matches the requirements for this job. This may not be serious since multiple passes may occur.
<b>0x210-0a787</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_GET_TASK_ON_RESERVATION	Cannot get tasks on (ERR: %s/no reservation/iteration %s).	Cannot select tasks that meet the requirements.
<b>0x210-0a78a</b>	ADMIN	system-moab	ERROR	MWM_BEST_VAL_ACHIEVED_BUT_SCHEDULE_EMPTY	BestVal %s achieved but schedule is empty.	Best value has been set, but the schedule is empty.
<b>0x210-0a78b</b>	ADMIN	system-moab	ERROR	MWM_JOB_SCHEDULING_FAILURE_NO_RESERVATION	Scheduling failure %s (policy violation/no reservation) iteration: %s. (%s)	The job was not scheduled because no reservations are available.
<b>0x210-0a78c</b>	ADMIN	system-moab	ERROR	MWM_UNSUPPORTED_SERVICE	Service '%s' (%s) not supported.	A request for an unsupported service was sent.
<b>0x210-0a78d</b>	ADMIN	system-moab	ERROR	MWM_INVALID_CLASS_HOST_EXPRESSION	Invalid class host expression received (%s) : %s.	Failed to expand the class's host pattern to a list.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a78e</b>	ADMIN	system-moab	ERROR	MWM_TOO_MANY_COALLOCATION_REQUESTS	Too many co-allocation requests (%s > %s).	Too many co-allocation requests were received.
<b>0x210-0a78f</b>	ADMIN	system-moab	ERROR	MWM_INVALID_JOBID_COUNTER	Min Job ID '%s' must be less than Max Job ID '%s'.	Invalid job id was encountered.
<b>0x210-0a790</b>	ADMIN	system-moab	ERROR	MWM_PARAMETER_NOT_HANDLED	Parameter[%s] '%s' not handled.	The specified parameter was not handled due to an unknown format.
<b>0x210-0a791</b>	ADMIN	system-moab	ERROR	MWM_CIRCULAR_JOB_DEPENDENCY	Job cannot be dependent on itself.	The job is trying to use itself as a dependency, which creates a circular dependency and is invalid.
<b>0x210-0a792</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_AM	Cannot create AM %s.	Could not create account manager object.
<b>0x210-0a793</b>	ADMIN	system-moab	ERROR	MWM_INVALID_FLUSH_INTERVAL	%s for AM %s.	An invalid flush interval has been entered.
<b>0x210-0a794</b>	ADMIN	system-moab	ERROR	MWM_FAILED_SERVER_AUTH	Unable to authenticate server.	The server could not be authenticated.
<b>0x210-0a795</b>	ADMIN	system-moab	ERROR	MWM_NO_QUOTE	No quote output provided in response.	No quote output provided in response.
<b>0x210-0a796</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_TO_PARSE_XML	Unable to parse XML (%s): %s.	Unable to parse XML data.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a797</b>	ADMIN	system-moab	ERROR	MWM_INVALID_QUOTE	Invalid quote amount (%s).	Quote is invalid.
<b>0x210-0a798</b>	ADMIN	system-moab	ERROR	MWM_RECURRING_COST	Unable to determine recurring cost.	Unable to determine recurring cost.
<b>0x210-0a799</b>	ADMIN	system-moab	ERROR	MWM_AVAILABLE_PORT_NOT_FOUND	Cannot locate an available port for listening.	After trying to bind to a large number of ports, none were found to be available. Check network socket status for saturation.
<b>0x210-0a79a</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_RESOLVE_IP_FROM_HOSTNAME	Cannot resolve IP address from host-name '%s', getaddrinfo() rc: %s (%s).	There is a failure matching an IP address to a host-name. Check DNS, /etc/hosts or applicable nameservice.
<b>0x210-0a79b</b>	ADMIN	system-moab	ERROR	MWM_UNKNOWN_CHECKPOINT_TYPE	Unexpected checkpoint type, %s.	Unknown checkpoint type while reading from the file.
<b>0x210-0a79c</b>	ADMIN	system-moab	ERROR	MWM_CHECKPOINT_FILE_LINE_NOT_HANDLED	Line '%s' not handled in checkpoint file '%s'.	Please contact Adaptive Computing for assistance.
<b>0x210-0a79d</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_ADD_DEFAULT_GROUP	Cannot add default group.	Default group cannot be added.
<b>0x210-0a79e</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_ADD_GROUP	Cannot add group %s.	Group cannot be added.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a79f</b>	ADMIN	system-moab	ERROR	MWM_ACCOUNT_NOT_ACCESSIBLE_BY_JOB	Account '%s' is not accessible by job '%s'.	The job is not authorized to run under the listed account.
<b>0x210-0a7a0</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_DETERMINE_DEFAULT_ACCOUNT	Unable to determine default account for job '%s', user '%s'.	There is not a default account type for this job.
<b>0x210-0a7a1</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_RESERVATION	Cannot create reservation for job '%s'.	Failed to create reservation for job.
<b>0x210-0a7a2</b>	ADMIN	system-moab	ERROR	MWM_INVALID_NODELIST_BAD_TASKCOUNT	Invalid nodelist for job %s:%s (inadequate taskcount, %s < %s).	Invalid node list due to inadequate taskcount.
<b>0x210-0a7a3</b>	ADMIN	system-moab	ERROR	MWM_INVALID_NODELIST_BAD_NODECOUNT	Invalid nodelist for job %s:%s (inadequate nodecount, %s < %s).	Invalid node list due to inadequate nodecount.
<b>0x210-0a7a4</b>	ADMIN	system-moab	ERROR	MWM_INVALID_ALLOCATION_POLICY	Invalid allocation policy (%s).	Invalid allocation policy.
<b>0x210-0a7a5</b>	ADMIN	system-moab	ERROR	MWM_NO_MEMORY_FOR_ALLOCPARTITION_VARIABLE	Cannot set ALLOCPARTITION variable for job %s (no memory).	No memory remaining to create job variable.
<b>0x210-0a7a6</b>	ADMIN	system-moab	ERROR	MWM_BASIL_RSVID_NOT_FOUND	Cannot locate BASIL RSVID (job 'ALLOCPARTITION' variable) that was just created.	Cannot locate BASIL reservation id stored in the ALLOCPARTITION variable.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7a7</b>	ADMIN	system.-moab	ERROR	MWM_JOB_ADD_CLASS_ATTR	Cannot add class for job %s (Class: %s).	Unable to add a class requirement attribute to a job.
<b>0x210-0a7a8</b>	ADMIN	system.-moab	ERROR	MWM_JOB_ADD_DRM_ATTR	Cannot set destination RM for job %s (RM: %s).	Unable to add a destination resource manager attribute to a job.
<b>0x210-0a7a9</b>	ADMIN	system.-moab	ERROR	MWM_JOB_FLAGS_INVALID_SOURCE	Attempting to set job flags from invalid format.	Job flags must be created using documented formats.
<b>0x210-0a7aa</b>	ADMIN	system.-moab	ERROR	MWM_SET_SIZE_ON_NONEXISTENT_REQ	Requirement must be created before size is set.	Unable to set the size of an unallocated requirement.
<b>0x210-0a7ab</b>	ADMIN	system.-moab	ERROR	MWM_ADD_GROUP_TO_JOB_FAILURE	Cannot add group for job %s (Group: %s).	Unable to set a group attribute on a job.
<b>0x210-0a7ac</b>	ADMIN	system.-moab	ERROR	MWM_NULL_JOB_NAME	Cannot add an empty name as an alternate name attribute for job %s.	No value specified. Make sure the alternate job name has a value.
<b>0x210-0a7ad</b>	ADMIN	system.-moab	ERROR	MWM_SPACES_IN_JOB_NAME	Attempted to set a job name (%s) with space(s) for job %s.	A job name with space(s) was specified. Job names cannot contain embedded spaces.
<b>0x210-0a7ae</b>	ADMIN	system.-moab	ERROR	MWM_ADD_QOS_TO_JOB_FAILURE	Cannot add QOS for job %s (QOS: %s).	Unable to set a QOS attribute on a job.
<b>0x210-0a7af</b>	ADMIN	system.-moab	ERROR	MWM_ADD_SRM_TO_JOB_FAILURE	Cannot add Submit RM for job %s (RM: %s).	Unable to find the entered name as an available resource manager.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7b0</b>	ADMIN	system-moab	ERROR	MWM_ADD_VARIABLE_TO_JOB_FAILURE	Cannot set variable for job %s (no variable name specified).	Only variables with names can be added as a job attribute.
<b>0x210-0a7b1</b>	ADMIN	system-moab	ERROR	MWM_ADD_USER_TO_JOB_FAILURE	Cannot add user for job %s (User: %s).	Unable to set a user attribute on a job.
<b>0x210-0a7b2</b>	ADMIN	system-moab	ERROR	MWM_ADD_NODE_TO_JOB_FAILURE	Cannot add node for job %s (Node: %s).	Unable to set a node attribute on a job.
<b>0x210-0a7b3</b>	ADMIN	system-moab	ERROR	MWM_ADD_ACCOUNT_TO_JOB_FAILURE	Cannot add account for job %s (Name: %s).	Failed to add account to the job.
<b>0x210-0a7b5</b>	ADMIN	system-moab	ERROR	MWM_INVALID_TIME_STRING	Invalid format for time specification: '%s'.	A string that describes a time cannot be parsed because the format is wrong, or the values are out of range.
<b>0x210-0a7b6</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_FIND_ARRAY_JOB	Cannot find array job at index %s for job '%s'.	Array job is missing.
<b>0x210-0a7b7</b>	ADMIN	system-moab	ERROR	MWM_JOB_BUFFER_FULL	Job buffer is full (ignoring job '%s').	Ignoring job since job buffer is full. Try increasing the value specified for the MAXJOB parameter.
<b>0x210-0a7b8</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_FIND_MASTER_JOB	Cannot find master job (%s) for job '%s'; job array slot limits may not be enforced.	Cannot find the master job that is associated with a job array.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7b9</b>	ADMIN	system.-moab	ERROR	MWM_INVALID_ACTION_STRING	The action string (%s) is invalid.	The format of the action string is '<operation type>:<-operation ID>:<-operation action>' Example: job:145+146+147:cancel where 145,146 and 147 are job IDs.
<b>0x210-0a7ba</b>	ADMIN	system.-moab	ERROR	MWM_INVALID_OBJECT_TYPE	The object type %s is invalid.	The format of the action string is '<operation type>:<-operation ID>:<-operation action>' Example: job:145+146+147:cancel where 145,146 and 147 are job IDs.
<b>0x210-0a7bb</b>	ADMIN	system.-moab	ERROR	MWM_JOB_NOT_FOUND	Unable to locate job %s.	The named job was not located in the system.
<b>0x210-0a7bc</b>	ADMIN	system.-moab	ERROR	MWM_JOB_IN_BAD_STATE_FOR_COMPLETE	Completed trigger action is specified for job %s but it is in an invalid state.	The job is not a system job and is not allowed to be started by the resource manager.
<b>0x210-0a7bd</b>	ADMIN	system.-moab	ERROR	MWM_JOB_CANNOT_BE_HELD	Job %s cannot be put into hold state.	The resource manager cannot hold the job, usually because the job is not in a state that can be held.
<b>0x210-0a7be</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_SET_TRIGVAR	Cannot set trigger variable on job %s.	The trigger variables on a job cannot be set.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
0x210-0a7bf	ADMIN	system-moab	ERROR	MWM_CANNOT_SET_REQATTR	Cannot set request attribute variable on job %s.	The request attribute variables on a job cannot be set.
0x210-0a7c0	ADMIN	system-moab	ERROR	MWM_CANNOT_ADJUST_GRES	Cannot adjust generic resources for job %s.	The generic resources of the job could not be modified.
0x210-0a7c1	ADMIN	system-moab	ERROR	MWM_INVALID_GRES_VALUE	Invalid value '%s' for GRes '%s' %s.	The value being set on the generic resource is not valid.
0x210-0a7c2	ADMIN	system-moab	ERROR	MWM_CANNOT_MODIFY_ATTRIBUTE	Attribute %s cannot be modified for job %s.	The job's attribute could not be modified.
0x210-0a7c3	ADMIN	system-moab	ERROR	MWM_COULD_NOT_SEND_SIGNAL	Signal %s could not be sent to job %s.	The resource manager was unable to send the signal to the job.
0x210-0a7c4	ADMIN	system-moab	ERROR	MWM_COULD_NOT_START_JOB	Could not start job %s in %s.	The resource manager was unable to start the job.
0x210-0a7c5	ADMIN	system-moab	ERROR	MWM_UNABLE_TO_REQUEUE_JOB	Cannot requeue job %s.	The job could not be requeued.
0x210-0a7c6	ADMIN	system-moab	ERROR	MWM_UNHANDLED_ACTION	The action %s was not handled.	The action was undefined in this function.
0x210-0a7c7	ADMIN	system-moab	ERROR	MWM_UNRECOGNIZED_ATTRIBUTE	The attribute %s is not recognized.	The attribute is not in the lookup table.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7c8</b>	ADMIN	system.-moab	ERROR	MWM_UNRECOGNIZED_JOB_ACTION	The job action %s is not recognized.	The job action is not in the lookup table.
<b>0x210-0a7c9</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_CANCEL_JOB	Job %s could not be canceled.	The job could not be canceled.
<b>0x210-0a7ca</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_HOLD_JOB	Job %s could not be held in.	The job was unable to be put into a hold state.
<b>0x210-0a7cb</b>	ADMIN	system.-moab	ERROR	MWM_INVALID_PBS_SBINDIR	Invalid SBINDIR specified (%s).	Check paths for the directory containing pbs_iff.
<b>0x210-0a7cc</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_CONNECT_PBS_SRVR	Cannot connect to PBS server '%s'; rc: %s (pbs_errno=%s, '%s').	Make sure the pbs_server process is running.
<b>0x210-0a7cd</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_GET_SRVR_INFO	Cannot get server info: %s.	Make sure that the pbs_server process is running.
<b>0x210-0a7ce</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_LOAD_SRVR_INFO	Cannot load PBS server info: %s.	Make sure that the pbs_server process is running.
<b>0x210-0a7cf</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_LOAD_PBS_CLUSTER	Cannot load PBS cluster info: %s.	Make sure that the pbs_server process is running.
<b>0x210-0a7d0</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_LOAD_PBS_WORKLOAD	Cannot load PBS workload info: %s.	Make sure that the pbs_server process is running.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7d1</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_LOAD_PBS_QUEUE	Cannot load PBS queue info: %s.	Make sure the path to the queue configuration is accessible by Moab.
<b>0x210-0a7d3</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_PROCESS_NODE_INFO	Cannot process node info.	Make sure the resource manager is running.
<b>0x210-0a7d4</b>	ADMIN	system-moab	ERROR	MWM_NODE_BUFFER_FULL	Node buffer is full (ignoring node '%s').	Try increasing the node buffer.
<b>0x210-0a7d5</b>	ADMIN	system-moab	ERROR	MWM_JOB_CANNOT_START	Job '%s' cannot be started: (cannot generate Tasklist).	Check the PBS server log to see reason of failure.
<b>0x210-0a7d6</b>	ADMIN	system-moab	ERROR	MWM_JOB_CANNOT_START_TASK_EMPTY	Job '%s' cannot be started: (empty Tasklist).	Check the PBS server log to see reason of failure.
<b>0x210-0a7d7</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_SET_NODE_COUNT	Cannot set nodecount for job '%s' - %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7d8</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_SET_WALLTIME	Cannot set walltime for job '%s' - %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7d9</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_SET_TASKLIST	Cannot set Tasklist for job '%s' - %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7da</b>	ADMIN	system-moab	ERROR	MWM_UNABLE_TO_START_JOB_RC	Job '%s' cannot be started: (rc: %s; errmsg: '%s'; Tasklist: '%s').	Check the PBS server log to see reason of failure.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7db</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_SIGNAL_JOB	%s' cannot be signaled: %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7dc</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_SUSPEND_JOB	Job '%s' cannot be suspended: %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7dd</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_RESUME_JOB	Job '%s' cannot be resumed: %s.	Check the PBS server log to see reason of failure.
<b>0x210-0a7de</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_FIND_RESOURCE	Failed to find/add %s generic resource.	Failure to find/add GPUs/MICs to the global GRES/MIC slots.
<b>0x210-0a7df</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_SET_CREDENTIALS	Cannot authenticate job '%s' (U: %s; G: %s; A: '%s').	Could not set the credentials on the job.
<b>0x210-0a7e0</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_REQUEUE	PBS job '%s' cannot be requeued (rc: %s; '%s').	Check the PBS server log to see reason of failure.
<b>0x210-0a7e1</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_CHECKPOINT	PBS job '%s' cannot be checkpointed (rc: %s; '%s').	Check the PBS server log to see reason of failure.
<b>0x210-0a7e2</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_RELEASE	PBS job '%s' cannot be released from hold (rc: %s; '%s').	Check the PBS server log to see reason of failure.
<b>0x210-0a7e5</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_FIND_ACCOUNT	Cannot find account for job %s (Name: %s).	Make sure the account exists.
<b>0x210-0a7e6</b>	ADMIN	system.-moab	ERROR	MWM_INVALID_ARGUMENT	Command '%s' args not handled.	An unsupported argument was used.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7e7</b>	ADMIN	system-moab	ERROR	MWM_INVALID_LOGDIR	LogDir '%s' is invalid.	Make sure that the path to the logs directory exists.
<b>0x210-0a7e8</b>	ADMIN	system-moab	ERROR	MWM_INVALID_SPOOLDIR	SpoolDir '%s' is invalid.	Make sure that the path to the spool directory exists.
<b>0x210-0a7e9</b>	ADMIN	system-moab	ERROR	MWM_INVALID_STATDIR	StatDir '%s' is invalid.	Make sure that the path to the stat directory exists.
<b>0x210-0a7ea</b>	ADMIN	system-moab	ERROR	MWM_INVALID_TOOLS DIR	ToolsDir '%s' is invalid.	Make sure that the path to the tools directory exists.
<b>0x210-0a7eb</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_DAT_FILE	Cannot create/modify dat file: '%s'.	Moab encountered an error creating the dat file.
<b>0x210-0a7ec</b>	ADMIN	system-moab	ERROR	MWM_FEATURE_NOT_AVAILABLE_IN_BUILD	The '%s' feature is not available in the build of Moab.	Moab can be configured with various features. The listed feature is not available in the binary being run.
<b>0x210-0a7ed</b>	ADMIN	system-moab	ERROR	MWM_FEATURE_NOT_AVAILABLE_WITH_LICENSE	The '%s' feature is not enabled with the current Moab license.	Moab can be licensed with various features. The listed feature is not available with the current license. Contact Adaptive Computing for more information.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7ee</b>	ADMIN	system.-moab	ERROR	MWM_RESOURCE_LIMIT_EXCEEDED	The maximum number of '%s' (%s) has been reached.	Moab has certain resources that are limited. This error occurs when you have reached or exceeded those limits. Contact Adaptive Computing for more information.
<b>0x210-0a7f1</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_CREATE_VM_MIGRATION_JOB	Failed to create migration job for VM %s.	The migration job was not created. Check MIGRATETEMPLATE on workflow and its trigger.
<b>0x210-0a7f2</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_OPEN_EXTENSION_INTERFACE	Cannot open extension interface socket on port %s.	There was a failure opening the HTTP extension service. This feature will not work until the problem is corrected.
<b>0x210-0a7f3</b>	ADMIN	system.-moab	ERROR	MWM_JOB_USER_AUTHENTICATION	The system was unable to connect the given user to job %s (User: %s, Group: %s).	Check the credentials of the given user and/or group.
<b>0x210-0a7f4</b>	ADMIN	system.-moab	ERROR	MWM_JOB_AUTHENTICATION	The system was unable to authenticate the user connected with job %s (User: %s, Group: %s, Account %s) - %s.	Check the credentials of the given user and/or group.
<b>0x210-0a7f5</b>	ADMIN	system.-moab	ERROR	MWM_SEND_DATA_FAILED	The system was unable to send data to the server %s (%s:%s).	Make sure the server's address is correct and that the server is running.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
0x210-0a7f6	ADMIN	system-moab	ERROR	MWM_RECEIVE_DATA_FAILED	The system was unable to receive data from the server %s (%s:%s).	Make sure the server's address is correct and that the server is running.
0x210-0a7f7	ADMIN	system-moab	ERROR	MWM_JOB_OVERLAP	Job '%s' overlaps an existing job.	Check the job being created for overlap.
0x210-0a7f8	ADMIN	system-moab	ERROR	MWM_JOB_CREATION	The system was unable to create job '%s'	Verify that the job being created is correctly specified.
0x210-0a7f9	ADMIN	system-moab	ERROR	MWM_MISSING_STATUS_ELEMENT	The status element was missing from the S3 response.	This is an internal error.
0x210-0a7fa	ADMIN	system-moab	ERROR	MWM_VC_WORKFLOW_JOB	Virtual container '%s' was marked as workflow, but could not find job that created it.	This is an internal error.
0x210-0a7fb	ADMIN	system-moab	ERROR	MWM_VC_COMBINE_JOBS	Failed to combine jobs in virtual container '%s'.	This is an internal error.
0x210-0a7fc	ADMIN	system-moab	ERROR	MWM_VC_SCHEDULE_TIME_FAILURE	Failed to schedule virtual container '%s' for requested time.	This is an internal error.
0x210-0a7fd	ADMIN	system-moab	ERROR	MWM_VC_RESERVATION_FAILURE	Failed to find a reservation for virtual container '%s'.	This is an internal error.
0x210-0a7fe	ADMIN	system-moab	ERROR	MWM_VC_RESERVATION_CREATE_FAILURE	Failed to create a reservation for jobs in virtual container '%s'.	This is an internal error.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a7ff</b>	ADMIN	system.-moab	ERROR	MWM_VC_RESOURCE_FAILURE	Requested resources are not available at any time for virtual container '%s'.	This is an internal error.
<b>0x210-0a800</b>	ADMIN	system.-moab	ERROR	MWM_NONEXISTING_JOB_USER	Job template %s requests non-existent user %s.	Make sure the user exists.
<b>0x210-0a801</b>	ADMIN	system.-moab	ERROR	MWM_NONEXISTING_JOB_GROUP	Job template %s requests non-existent group %s.	Make sure the group exists.
<b>0x210-0a802</b>	ADMIN	system.-moab	ERROR	MWM_NONEXISTING_JOB_QOS	Job template %s requests non-existent QoS %s.	Make sure the QoS exists.
<b>0x210-0a803</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_CREATE_CLASS	Unable to create class %s for job template %s.	Make sure the class exists.
<b>0x210-0a804</b>	ADMIN	system.-moab	ERROR	MWM_NONEXISTING_JOB_ACCOUNT	Job template %s requests non-existent account %s.	Make sure the account exists.
<b>0x210-0a805</b>	ADMIN	system.-moab	ERROR	MWM_INVALID_WALLTIME_SPECIFIED	Invalid walltime specification '%s'.	Make sure the format for walltime is correct.
<b>0x210-0a806</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_PARSE_WIKI_STR	Cannot parse wiki string for job '%s'.	Make sure the format for wiki string is correct.
<b>0x210-0a807</b>	ADMIN	system.-moab	ERROR	MWM_MISSING_STATS_XML_ELEMENT	%s is not a valid template job stat child element.	Make sure there is a stats element in the XML.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a808</b>	ADMIN	system-moab	ERROR	MWM_NULL_NODE_POINTER	Node pointer is NULL and cannot be used to find SMP node.	Node pointer is NULL and cannot be used to find SMP node by node.
<b>0x210-0a809</b>	ADMIN	system-moab	ERROR	MWM_PINDEX_OUT_OF_RANGE	PIndex is less than -1 which is out of range.	PIndex must be greater than or equal to -1 to find a node by partition.
<b>0x210-0a80a</b>	ADMIN	system-moab	ERROR	MWM_FEATURE_OUT_OF_RANGE	Feature is less than -1 which is out of range.	Feature must be greater than or equal to -1 to find a node by feature.
<b>0x210-0a80b</b>	ADMIN	system-moab	ERROR	MWM_INCORRECT_ARG	Incorrect argument in %s: %s, %s, %s.	Name must point to a valid string, Feature must be greater than or equal to -1, and N must point to a valid node.
<b>0x210-0a80c</b>	ADMIN	system-moab	ERROR	MWM_NODE_ALLOCATION_ERROR	Failed to allocate a node named %s.	Call to MUMalloc failed, system is probably low on memory.
<b>0x210-0a80d</b>	ADMIN	system-moab	ERROR	MWM_FAILED_TO_APPEND_MSMPNODE	Failed to append smpnode %s to MSMPNodes.	The call to append the node to the array list failed, probably due to a low memory condition.
<b>0x210-0a80e</b>	ADMIN	system-moab	ERROR	MWM_NULL_SMPNODE_POINTER	Cannot initialize node because pointer is NULL.	Call to MSMPNodeInitialize must have a valid pointer to a valid node.
<b>0x210-0a80f</b>	ADMIN	system-moab	ERROR	MWM_NULL_SMPNODE_POINTER_IN_RESET	Cannot reset node because pointer is NULL.	Call to MSMPNodeResetStats must have a valid pointer to a valid node.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a810</b>	ADMIN	system.-moab	ERROR	MWM_RESET_NODE_FAILED	Call to MSMPNodeResetStats failed.	Call to MSMPNodeResetStats failed. The most likely cause is passing a NULL pointer to SMPNode.
<b>0x210-0a811</b>	ADMIN	system.-moab	ERROR	MWM_FREE_NODE_FAILED	Call to free MSMPNodes failed.	Call to free MSMPNodes failed, most likely due to corrupted memory.
<b>0x210-0a812</b>	ADMIN	system.-moab	ERROR	MWM_NULL_NODE_IN_UPDATE	Node pointer in %s cannot be NULL.	Node pointer cannot be NULL when trying to update node.
<b>0x210-0a813</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_FIND_NODE	Unable to find SMP node with node %s.	Unable to find SMP node by node.
<b>0x210-0a814</b>	ADMIN	system.-moab	ERROR	MWM_EMPTY_NODE_LIST	Updating node from list with empty node list.	Updating node from node list must not be called with an empty node list.
<b>0x210-0a815</b>	ADMIN	system.-moab	ERROR	MWM_BAD_ARG_IN_FEASIBLE_JOB	Incorrect argument to function %s: %s, %s.	A parameter in the function was incorrect.
<b>0x210-0a816</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_FIND_INDEX_IN_LIST_FOR_FEATURE	Could not find index into NodeSetList for node feature %s.	Could not find index into NodeSetList for node feature.
<b>0x210-0a817</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_CREATE_SR	Could not create standing reservation: %s.	Failed to create the named standing reservation.
<b>0x210-0a818</b>	ADMIN	system.-moab	ERROR	MWM_UNEXPECTED_STATISTICS_TYPE	Unexpected statistics type: %s.	Number is not a member of MMStatTypeEnum enumeration.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a819</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_PROCESS_VM_ATTRIBUTE	Cannot process VM attribute %s for VM %s.	Either AttrName or NodeName is not found in string.
<b>0x210-0a81a</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_FIND_NODE_FOR_VM	Cannot find node %s for VM %s.	The node does not exist or cannot be found.
<b>0x210-0a81b</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_LOAD_JOB	Cannot load job %s (state: %s).	There was an error creating a job in Moab that was reported by the resource manager.
<b>0x210-0a81c</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_CHECKPOINT_FILE_ENTRY	Cannot create checkpoint file entry.	There was an error writing a checkpoint file entry for the associated objects.
<b>0x210-0a81d</b>	ADMIN	system-moab	ERROR	MWM_CANNOT_CREATE_OBJECT_FROM_CHECKPOINT_FILE	Cannot create object from checkpoint file entry.	There was an error reading a checkpoint file entry for the associated objects.
<b>0x210-0a81e</b>	ADMIN	system-moab	ERROR	MWM_TASKLIST_TOO_LARGE	The tasklist for job '%s' is too large (size = %s, growth = %s).	The system has a fixed maximum size for the task map for each job.
<b>0x210-0a81f</b>	ADMIN	system-moab	ERROR	MWM_TASKLIST_MISSING	The tasklist for job '%s' is missing.	The system requires that each job has at least one task assigned.
<b>0x210-0a820</b>	ADMIN	system-moab	ERROR	MWM_TASK_DISTRIBUTION_UNKNOWN	The system encountered an unknown type of task distribution (%s).	This is an internal error.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a822</b>	ADMIN	system.-moab	ERROR	MWM_INCOMPATIBLE_CHARGE_POLICY	Periodic charging disabled due to incompatible charge policy (%s).	The charge policy is undefined.
<b>0x210-0a823</b>	ADMIN	system.-moab	ERROR	MWM_INCOMPLETE_JOB_TEMPLATE_ACTION	The job template '%s' has an incomplete action specification.	Job templates must fully specify the action to be performed.
<b>0x210-0a824</b>	ADMIN	system.-moab	ERROR	MWM_INCOMPLETE_JOB_TEMPLATE_GENERIC	The job template '%s' has an incomplete generic system job specification.	Job templates must fully specify the generic system job.
<b>0x210-0a825</b>	ADMIN	system.-moab	ERROR	MWM_DUPLICATE_JOB_TEMPLATE_VMID	The job template '%s' has a job '%s' that requests an existing VMID.	Virtual machine IDs cannot be shared across job templates.
<b>0x210-0a826</b>	ADMIN	system.-moab	ERROR	MWM_UNKNOWN_JOB_TEMPLATE_VMID	The requested VMID '%s' could not be found or already has a tracking job.	Virtual machine IDs can only be assigned to a single job.
<b>0x210-0a827</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_MODIFY_JOB	The job '%s' on account '%s' cannot be modified in the resource manager.	The job previously submitted to the resource manager cannot be modified.
<b>0x210-0a828</b>	ADMIN	system.-moab	ERROR	MWM_WORKFLOW_VC_FAILURE	The system failed to generate a workflow virtual container for job '%s'.	This is an internal error.
<b>0x210-0a829</b>	ADMIN	system.-moab	ERROR	MWM_CREATE_JOB_TEMPLATE_FAILURE	The system failed to create job template '%s'.	The job could not be created or one of its attributes could not be set.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a82a</b>	ADMIN	system-moab	ERROR	MWM_VC_NOT_FOUND	The system could not find the virtual container for job '%s'.	This is an internal error.
<b>0x210-0a82b</b>	ADMIN	system-moab	ERROR	MWM_JOB_MIGRATION_FAILED	The system failed to migrate a remote job (%s).	Make sure the resource manager has not been disabled.
<b>0x210-0a82c</b>	ADMIN	system-moab	ERROR	MWM_JOB_START_XML_FAILURE	The system could not generate the command line needed to start job: '%s'.	The proper command line could not be derived from the XML structure.
<b>0x210-0a82d</b>	ADMIN	system-moab	ERROR	MWM_JOB_START_FAILURE_RESPONSE	The system could not start job - Reason: '%s'.	The system was unable to start the job for the specified reason.
<b>0x210-0a82e</b>	ADMIN	system-moab	ERROR	MWM_JOB_CANCEL_FAILURE_RESPONSE	The system could not cancel job - Reason: '%s'.	The system was unable to cancel the job for the specified reason.
<b>0x210-0a82f</b>	ADMIN	system-moab	ERROR	MWM_JOB_SIGNAL_FAILURE_RESPONSE	The system could not signal job - Reason: '%s'.	The system was unable to signal the job for the specified reason.
<b>0x210-0a830</b>	ADMIN	system-moab	ERROR	MWM_JOB_MODIFY_FAILURE_RESPONSE	The system could not modify job - Reason: '%s'.	The system was unable to modify the job for the specified reason.
<b>0x210-0a831</b>	ADMIN	system-moab	ERROR	MWM_JOB_REQUEUE_FAILURE_RESPONSE	The system could not requeue job - Reason: '%s'.	The system was unable to requeue the job for the specified reason.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a832</b>	ADMIN	system.-moab	ERROR	MWM_SEND_EVENT_FAILURE	The system could send event '%s' to resource manager '%s' (%s).	The system was unable to send the event.
<b>0x210-0a833</b>	ADMIN	system.-moab	ERROR	MWM_UNEXPECTED_BACKFILL_POLICY	The system encountered an unexpected backfill policy '%s' (using '%s' instead).	The backfill policy did not match a defined policy.
<b>0x210-0a834</b>	ADMIN	system.-moab	ERROR	MWM_NODE_LIST_ALLOCATION	The system was unable to allocate a node list for job '%s' in partition '%s'.	The system may be low on memory.
<b>0x210-0a835</b>	ADMIN	system.-moab	ERROR	MWM_BAD_NODE_IN_NODELIST	The reservation nodelist for job '%s' has an invalid node at index %s.	Check the nodes specified for the reservation.
<b>0x210-0a836</b>	ADMIN	system.-moab	ERROR	MWM_RESERVATION_SPANS_PARTITIONS	The reservation request for job '%s' spans partitions (node %s partition %s).	Reservations that span partitions must have the COALLOC flag set.
<b>0x210-0a837</b>	ADMIN	system.-moab	ERROR	MWM_ADJUST_JOB_RESERVATION_FAILURE	The system failed to adjust job '%s' reservation on node %s.	This is an internal error.
<b>0x210-0a838</b>	ADMIN	system.-moab	ERROR	MWM_OBJECT_TYPE_INVALID	The object type specified (%s) is not valid.	A valid object type must be specified.
<b>0x210-0a839</b>	ADMIN	system.-moab	ERROR	MWM_MISSING_OBJECT_ID	The object ID is missing.	A valid object ID must be specified.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a83a</b>	ADMIN	system-moab	ERROR	MWM_MISSING_ACTION	The action is missing.	A valid action must be specified.
<b>0x210-0a83b</b>	ADMIN	system-moab	ERROR	MWM_PIPE_BUFFER_FAILED	The system could not open a bi-directional pipe.	A valid action must be specified.
<b>0x210-0a83c</b>	ADMIN	system-moab	ERROR	MWM_STD_OUT_FAILED	Failed to load stdout file '%s'.	Check the file name and path.
<b>0x210-0a83d</b>	ADMIN	system-moab	ERROR	MWM_STD_ERR_FAILED	Failed to load stderr file '%s'.	Check the file name and path.
<b>0x210-0a841</b>	ADMIN	system-moab	ERROR	MWM_CREATE_NODE_FAILURE	The system was unable to create node '%s'.	The system may be low on memory.
<b>0x210-0a842</b>	ADMIN	system-moab	ERROR	MWM_PARTITION_CREATE_FAILURE	The system was unable to create a shared partition for the global node.	The system may be low on memory.
<b>0x210-0a845</b>	ADMIN	system-moab	ERROR	MWM_HT_FIND_NODE_FAILURE	Cannot find node '%s' in hash table.	A node by the given name may not have been created.
<b>0x210-0a847</b>	ADMIN	system-moab	ERROR	MWM_HT_FIND_VM_FAILURE	Cannot find VM '%s' in hash table.	A VM with the given name may not have been created.
<b>0x210-0a848</b>	ADMIN	system-moab	ERROR	MWM_COMMAND_FAILED	Command '%s' failed. StatusCode: %s; Response: '%s'.	Check the command syntax and parameters.
<b>0x210-0a849</b>	ADMIN	system-moab	ERROR	MWM_HASH_TABLE_INITIALIZATION	There was an unexpected hash table initialization error.	The hash table for jobs to delete never initialized correctly.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a84e</b>	ADMIN	system.-moab	ERROR	MWM_UNABLE_TO_AUTHENTICATE_JOB	Unable to authenticate job %s when UID or GID is empty (UID=%s, GID=%s).	Either the UID or the GID field is empty.
<b>0x210-0a84f</b>	ADMIN	system.-moab	ERROR	MWM_MISSING_JOB_TASKCOUNT	Job does not have a taskcount specified.	Each job must have an associated taskcount.
<b>0x210-0a850</b>	ADMIN	system.-moab	ERROR	MWM_FAILED_EXCLUDE_NODELIST	The system failed to add an exclude nodelist to a submission.	The job exclude hostlist could not be converted into a string.
<b>0x210-0a852</b>	ADMIN	system.-moab	ERROR	MWM_CANNOT_SUBMIT_VM_MIGRATION_JOB	Failed to submit migration job for VM %s.	Check MIGRATETEMPLATE on workflow and its trigger.
<b>0x210-0a853</b>	ADMIN	system.-moab	ERROR	MWM_WEB_SERVICES_WRITE_FAILURE	Error %s encountered while trying to write to web services.	Encountered problem trying to put HTTP data to web server.
<b>0x210-0a854</b>	ADMIN	system.-moab	ERROR	MWM_WEB_SERVICES_URL_MISSING	Missing URL in call to web services.	Web services must have a valid destination URL.
<b>0x210-0a855</b>	ADMIN	system.-moab	ERROR	MWM_RM_PARTITION_CREATE_FAILURE	The system was unable to create a partition for RM '%s'.	The system may be low on memory.
<b>0x210-0a856</b>	ADMIN	system.-moab	ERROR	MWM_PARSE_MPP_NODES_FAILURE	The system failed to parse the MPP nodes value '%s'.	Check the MPP names.
<b>0x210-0a857</b>	ADMIN	system.-moab	ERROR	MWM_FIND_MPP_NODES_FAILURE	The system failed to find node '%s' in the MPP nodes value '%s'.	Check the MPP names.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a859</b>	ADMIN	system-moab	ERROR	MWM_NODE_SET_TYPE_INVALID	The node set type specified (%s) is not valid.	Check the NODESETLIST option.
<b>0x210-0a85a</b>	ADMIN	system-moab	ERROR	MWM_GRES_ADD_FAILURE	Unable to add the GRESTOJOBATTRMAP '%s'.	The limit has been reached.
<b>0x210-0a85d</b>	ADMIN	system-moab	ERROR	MWM_NOT_MWS_RM	The resource manager is not Moab Web Services.	Make sure the resource manager has Moab Web Services.
<b>0x210-0a85e</b>	ADMIN	system-moab	ERROR	MWM_MWS_RM_CURL_CONNECTION	The system could not initialize a cURL connection to the MWS RM.	The cURL command to connect to the resource manager has failed.
<b>0x210-0a85f</b>	ADMIN	system-moab	ERROR	MWM_MWS_RM_CURL_CONNECTION_EXPANDED	Could not connect to MWS RM (%s) at '%s%s' as '%s', response code: %s; cURL error: %s (%s); MWS response: '%s'.	The connection has failed.
<b>0x210-0a860</b>	ADMIN	system-moab	ERROR	MWM_MWS_RM_JSON_CLUSTER_QUERY_EMPTY	JSON cluster query data from MWS RM (%s) is null or empty.	The query must contain valid JSON data.
<b>0x210-0a861</b>	ADMIN	system-moab	ERROR	MWM_MWS_RM_JSON_WORKLOAD_QUERY_EMPTY	JSON workload query data from MWS RM (%s) is null or empty.	The query must contain valid JSON data.
<b>0x210-0a867</b>	ADMIN	system-moab	ERROR	MWM_JOB_TRANSITION_FAILURE	Unable to transition a job.	The job was missing requirements.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a868</b>	ADMIN	system.-moab	ERROR	MWM_SET_JOB_VARIABLE	Unable to set a job pref variable.	The system is probably low on memory.
<b>0x210-0a869</b>	ADMIN	system.-moab	ERROR	MWM_ARRAY_EXPANSION	Unable to expand the size of an array.	The system is probably low on memory.
<b>0x210-0a86c</b>	ADMIN	system.-moab	ERROR	MWM_VC_FIND_FAILURE	The system could not find the virtual container '%s'.	Check the name of the VC.
<b>0x210-0a86d</b>	ADMIN	system.-moab	ERROR	MWM_VC_USER_CREDENTIALS	User '%s' does not have access to virtual container '%s'.	Check the rights granted to the VC.
<b>0x210-0a86e</b>	ADMIN	system.-moab	ERROR	MWM_VC_BEING_DELETED	Virtual container '%s' is being deleted; cannot add jobs to it.	Only add jobs to VCs that are not being deleted.
<b>0x210-0a86f</b>	ADMIN	system.-moab	ERROR	MWM_PARTITION_STATUS	Unable to query the status of a partition - %s.	Check to make sure the resource manager is running.
<b>0x210-0a870</b>	ADMIN	system.-moab	ERROR	MWM_FIND_JOB_TEMPLATE	The system failed to find job template '%s'.	Check the template name for the given job.
<b>0x210-0a873</b>	ADMIN	system.-moab	ERROR	MWM_PROCESS_EVENT	Unable to process the generic event.	During processing, unable to get a description of the event.
<b>0x210-0a876</b>	ADMIN	system.-moab	ERROR	MWM_JOB_CAN_NEVER_RUN	Unable to allocate tasks for job at any time.	Job tasks must match available resources.
<b>0x210-0a877</b>	ADMIN	system.-moab	ERROR	MWM_NODE_NOT_IN_PARTITION	Node is not associated with any partition.	Node must be in a partition.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0a87a</b>	ADMIN	system-moab	ERROR	MWM_NODE_COUNT_EXCEEDS_LICENSE	The number of nodes '%s' exceeds the current license limit '%s'.	A different license is needed to use more nodes.
<b>0x210-0a87b</b>	ADMIN	system-moab	ERROR	MWM_KILL_FAILURE	OS call to kill process (PID: %s) %s failed).	This is an operating system error.
<b>0x210-0a87c</b>	ADMIN	system-moab	ERROR	MWM_MISSING_JOB_REQUIREMENTS	Job does not have any requirements specified.	Each job must have requirements attached.
<b>0x210-0a87d</b>	ADMIN	system-moab	ERROR	MWM_JOB_MISSING_DISPATCH_TIME	Job loaded in alloc state '%s' with no dispatch time.	The job must have a dispatch time.
<b>0x210-0a87e</b>	ADMIN	system-moab	ERROR	MWM_UNEXPECTED_OBJECT_TYPE	The object type '%s' was not expected in this operation.	Verify that a valid object type is given.
<b>0x210-0a87f</b>	ADMIN	system-moab	ERROR	MWM_JOB_TRANSITION_XML	Unable to create XML element from job transition object.	The system may be low on memory.
<b>0x210-0a881</b>	ADMIN	system-moab	ERROR	MWM_VM_CREATE_RESERVATION	Cannot create reservation for VM '%s'.	Failed to create reservation for the given VM.
<b>0x210-0aa08</b>	ADMIN	system-moab	ERROR	MWM_VM_FIELD_VALUE	VM '%s' has an invalid '%s%s%s' field value.	The field value for the VM is invalid.
<b>0x210-0aa09</b>	ADMIN	system-moab	ERROR	MWM_NODE_FIELD_VALUE	Node '%s' has an invalid '%s%s%s' field value.	The field value for the node is invalid.
<b>0x210-0aa0a</b>	ADMIN	system-moab	ERROR	MWM_JOB_FIELD_VALUE	Job '%s' has an invalid '%s%s%s' field value.	The field value for the job is invalid.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x210-0c001</b>	INTERNAL	system.-moab	ERROR	MWM_NOT_IMPLEMENTED	Function %s has not been implemented yet.	This error is used when we've stubbed out code but do not expect it to be called in production environments. It's not helpful except for internal diagnostics.
<b>0x210-0e729</b>	INTERNAL	system.-moab	ERROR	MWM_CANNOT_SEND_TO_SOCKET_DETAILED	Cannot send %s of %s bytes to socket descriptor %s - errno: %s (%s).	The send() system call failed. Socket is blocked (select() indicated socket was available--check MTU).
<b>0x210-0e77b</b>	INTERNAL	system.-moab	ERROR	MWM_CLIENT_COUNT_NEGATIVE	Client count fell below zero on socket %s.	This is an internal error. The number of client connections should always be zero or greater.
<b>0x210-0e784</b>	INTERNAL	system.-moab	ERROR	MWM_HOSTLIST_MISSING	A hostlist was specified but now it is NULL/EMPTY.	The job claims to have a specified hostlist, but at the current point in processing no list can be found. This is most likely an internal problem.
<b>0x210-0e7b4</b>	INTERNAL	system.-moab	ERROR	MWM_REQATTR_UNSUPPORTED_OPERATION	Operation (%s) not supported on required attributes (reqattrs).	See documentation for supported operators allows on required attributes (reqattrs).
<b>0x210-0e7ef</b>	INTERNAL	system.-moab	ERROR	MWM_VM_NOT_LINKED_TO_TRACKING_JOB	VM '%s' not linked to VMTracking job '%s' (linked to job '%s').	A VM must be associated with a tracking job.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x310-02a2c</b>	USER	system-moab	ALERT	MWM_NO_TASKS_FOUND_ON_JOB	No tasks found for job '%s'.	Check job submission arguments for desired requirements.
<b>0x310-08385</b>	ADMIN	system-moab	FATAL	MWM_EXPIRED_LICENSE	%s License has expired.	A license file was found but it has expired. Please contact your sales representative at Adaptive Computing for assistance.
<b>0x310-08386</b>	ADMIN	system-moab	FATAL	MWM_EVALUATION_EXPIRED	%s evaluation period has expired.	The evaluation period has expired. Please contact your sales representative at Adaptive Computing for assistance.
<b>0x310-08387</b>	ADMIN	system-moab	FATAL	MWM_UNEXPECTED_LICENSE_ERROR	Moab will now exit. Unexpected error while reading license: %s	Moab was unable to verify that the license file was valid. Please contact your sales representative at Adaptive Computing for assistance.
<b>0x310-0a712</b>	ADMIN	system-moab	FATAL	MWM_UNABLE_TO_ALLOCATE_MEMORY	Unable to allocate memory.	One or more calls to allocate memory failed.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x310-0a714</b>	ADMIN	system.-moab	FATAL	MWM_CANNOT_RESTORE_UID	Cannot restore EUID to '%s' for server, errno: %s (%s).	The setuid() system call failed. There was a failure resetting the UID of the process. This may be because the process is running as a different user. Use the errno and associated message to determine possible causes.
<b>0x310-0a715</b>	ADMIN	system.-moab	FATAL	MWM_CANNOT_CHANGE_UID	Cannot change UID to user '%s' (UID: %s) errno: %s (%s).	The setuid() system call failed. Use the errno and associated message to determine possible causes.
<b>0x310-0a716</b>	ADMIN	system.-moab	FATAL	MWM_CANNOT_RESTORE_GID	Cannot restore GID to '%s' for server, errno: %s (%s).	The setgid() system call failed. There was a failure resetting the GID of the process. This may be because the process is running in a different group. Use the errno and associated message to determine possible causes.
<b>0x310-0a717</b>	ADMIN	system.-moab	FATAL	MWM_CANNOT_FORK_INTO_BACKGROUND	Cannot fork the process into the background, errno: %s (%s).	The fork() system call failed. Moab must do this to daemonize unless run with the '-d' flag. This is usually due to low system resources.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x310-0a71c</b>	ADMIN	system-moab	FATAL	MWM_CANNOT_CHANGE_OWNERSHIP_FILE_FATAL	Cannot change ownership of %s file to uid:%s gid:%s errno: %s (%s).	The fchown() system call failed. Use the errno and associated message to determine possible causes.
<b>0x310-0a745</b>	ADMIN	system-moab	FATAL	MWM_CANNOT_GET_SERVER_HOSTNAME	Cannot determine hostname and attribute '%s' of parameter %s is not specified.	Moab failed to obtain system host name or ip address information from the operating system.
<b>0x310-0a746</b>	ADMIN	system-moab	FATAL	MWM_HA_MOAB_NOT_STARTED_ON_CORRECT_HOSTS	The server must be started on host '%s' or on alternate '%s' (currently on '%s').	Moab must be started on either the primary or alternate host for high availability.
<b>0x310-0a747</b>	ADMIN	system-moab	FATAL	MWM_MOAB_NOT_STARTED_ON_CORRECT_HOST	The server must be started on host '%s' (currently on '%s').	Moab must be started on specified host as identified by the SCHEDCFG parameter.
<b>0x310-0a749</b>	ADMIN	system-moab	FATAL	MWM_MOAB_ALREADY_RUNNING	Moab is already running. Cannot open user interface socket on port %s.	Cannot open user interface socket, which is most likely caused by Moab already running.
<b>0x310-0a74b</b>	ADMIN	system-moab	FATAL	MWM_CANNOT_LOCATE_FULL_PATH	Cannot locate the full path for '%s'.	Check the path to make sure the Moab executable is in it. Restart manually to work around this problem temporarily.
<b>0x310-0a74c</b>	ADMIN	system-moab	FATAL	MWM_CANNOT_RESTART_SCHEDULER	Exec failed when attempting to restart the scheduler '%s' rc: %s.	Please check permissions on this executable to correct and restart manually to work around.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x310-0a750</b>	ADMIN	system.-moab	FATAL	MWM_CANNOT_CONNECT_TO_DB_WITH_STRICT_CONFIG_CHECK_ON	StrictConfigCheck ON and cannot connect to DB--please check DB engine and configuration (%s).	Moab was unable to connect to the database and with strict configuration on Moab must exit.
<b>0x310-0a751</b>	ADMIN	system.-moab	FATAL	MWM_USER_NOT_AUTHORIZED_TO_RUN_THIS_PROGRAM	The user '%s' (UID: %s) is not authorized to run this program.	The user has insufficient privileges to run the program.
<b>0x310-0a752</b>	ADMIN	system.-moab	FATAL	MWM_PROBLEMS_WITH_KEY_FILE	Problems with key file.	Key file does not exist or ownership of key file is invalid.
<b>0x310-0aa0d</b>	ADMIN	system.-moab	FATAL	MWM_STRICT_CHECK_EXIT	Exiting because of strict configuration check.	Moab is configured to exit if there are any errors in configuration files or file/directory layout. One of these errors has occurred.
<b>0x310-0c002</b>	INTERNAL	system.-moab	FATAL	MWM_TESTING_FATAL	Testing with single argument: %s.	Internal error for testing diagnostics.
<b>0x310-0e74f</b>	INTERNAL	system.-moab	FATAL	MWM_CORRUPT_CHECKPOINT_FILE	Unable to read the checkpoint file.	Please contact Adaptive Computing for assistance.

## MWS Event Dictionary

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x02000100</b>	USER		INFO	Service Create	The service '{0}' was created	This marks when a service was created.
<b>0x02000101</b>	USER		INFO	Service Modify	The service '{0}' was modified	This marks when a service was modified.
<b>0x02000102</b>	USER		INFO	Service Transition	The service '{0}' took the '{1}' transition. It went from the '{2}' to the '{3}' phase.	This marks a service phase transition.
<b>0x02000103</b>	USER		INFO	Service Terminate	The service '{0}' was terminated	Service termination means that the resources are released and no more modifications may be made to the service or policies. However, it still resides in the database and shows as "Terminated".
<b>0x02000104</b>	USER		INFO	Service Delete	The service '{0}' was deleted and is no longer available	Service deletion occurs after termination and means that the service is fully removed from the database and will no longer be displayed in any queries to MWS.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x02000380</b>	USER		INFO	Notification Condition Create (User)	The notification condition '{0}' was created: {1}	This marks when a notification condition was created at the user escalation level.
<b>0x02000500</b>	USER		INFO	Service Template Create	The service template '{0}' was created	This marks when a service template was created.
<b>0x02000501</b>	USER		INFO	Service Template Modify	The service template '{0}' was modified	This marks when a service template was modified.
<b>0x02000502</b>	USER		INFO	Service Template Delete	The service template '{0}' was deleted and is no longer available	This marks when a service template was deleted.
<b>0x02000580</b>	USER		INFO	Service Hook Start	The service hook definition '{0}' on service '{1}' was executed with an ID of '{2}'	This marks the execution of a service hook definition for a given service.
<b>0x02000581</b>	USER		INFO	Service Hook End	The running service hook '{0}' on service '{1}' finished execution with status '{2}': {3}	This marks the end of execution for a running service hook.
<b>0x02000582</b>	USER		INFO	Service Hook Timeout	The running service hook '{0}' on service '{1}' timed out after {2} seconds.	This marks the execution of a service hook definition for a given service.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
0x02000583	USER		INFO	Service Hook Error	There was an error running service hook definition '{0}' on service '{1}' with an ID of '{2}'	This signifies that internal service hook processing has failed and the service hook could not be run successfully.
0x02004080	POWER_USER		INFO	Policy Modify	The policy '{0}' was modified	A policy modification occurs when a PUT client request is received by Moab Web Services for the policy resource.
0x02004381	POWER_USER		INFO	Notification Condition Create (Power User)	The notification condition '{0}' was created: {1}	This marks when a notification condition was created at the power user escalation level.
0x02008200	ADMIN		INFO	Permission Create	The permission '{0}' was created	This marks when a permission was created.
0x02008201	ADMIN		INFO	Permission Delete	The permission '{0}' was deleted	This marks when a permission was deleted.
0x02008280	ADMIN		INFO	Principal Create	The principal '{0}' was created	This marks when a principal was created.
0x02008281	ADMIN		INFO	Principal Modify	The principal '{0}' was modified	This marks when a principal was modified.
0x02008282	ADMIN		INFO	Principal Delete	The principal '{0}' was deleted	This marks when a principal was deleted.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
<b>0x02008300</b>	ADMIN		INFO	Role Create	The role '{0}' was created	This marks when a role was created.
<b>0x02008301</b>	ADMIN		INFO	Role Modify	The role '{0}' was modified	This marks when a role was modified.
<b>0x02008302</b>	ADMIN		INFO	Role Delete	The role '{0}' was deleted	This marks when a role was deleted.
<b>0x02008382</b>	ADMIN		INFO	Notification Condition Create (Admin)	The notification condition '{0}' was created: {1}	This marks when a notification condition was created at the administrator escalation level.
<b>0x02008400</b>	ADMIN		INFO	Tenant Create	The tenant '{0}' was created	This marks when a tenant was created.
<b>0x02008401</b>	ADMIN		INFO	Tenant Modify	The tenant '{0}' was modified	This marks when a tenant was modified.
<b>0x02008402</b>	ADMIN		INFO	Tenant Delete	The tenant '{0}' was deleted	This marks when a tenant was deleted.
<b>0x22004180</b>	POWER_USER		ERROR	LDAP Connect	Error communicating with the LDAP server: {0}	This occurs when communication could not be established with the configured LDAP server.

Code	Escalation level	Topic	Severity	Event name	Message template	Comment
0x22008105	ADMIN		ERROR	Service Transitions Missed	Service '{0}' ('{1}') is in phase '{2}' but should be in phase '{3}' according to the state reported by MWM. It is probable that one or more phase transitions were missed. Any hooks associated with those transitions were probably not executed.	This happens when the phase of a service as recorded by MWS differs from the phase expected according to the state of the service's jobs as reported by MWM. This means that one or more phase transitions were missed and thus any hooks on those transitions did not execute. This problem is most likely caused by either the message queue being misconfigured or MWS being down while MWM is running.
0x22008480	ADMIN		ERROR	Health Retrieval Failed	The health summary could not be retrieved successfully: {1}	This means a serious error occurred while attempting to retrieve the contents of the health summary REST resource from MWS.
0x22008481	ADMIN		ERROR	Health Create Failed	The notification condition for the failed health check could not be created: {1}	This marks when a notification condition failed to be created based on a health check.

## Appendix D: Adjusting Default Limits

Moab is distributed in a configuration capable of supporting multiple architectures and systems ranging from a few processors to several thousand processors. However, in spite of its flexibility, for performance reasons, it still contains a number of default object limits parameters and static structures defined in header files. These limits constrain such things as the maximum number of jobs, reservations, and nodes that Moab can handle and are set to values that provide a reasonable compromise between capability and memory consumption for most sites. However, many site administrators want to increase some of these settings to extend functionality, or decrease them to save consumed memory. The most common parameters are listed in what follows. Parameters listed in the Moab configuration file (`moab.cfg`) can be modified by restarting Moab. To change parameters listed in `moab.h`, please contact technical support.

### CLIENTMAXCONNECTIONS

<b>Location</b>	<code>moab.cfg</code> (dynamic parameter)
<b>Default</b>	128
<b>Max tested</b>	---
<b>Description</b>	Maximum number of connections that can simultaneously connect to Moab.

### JOBMAXNODECOUNT

<b>Location</b>	<code>moab.cfg</code> (dynamic parameter)
<b>Default</b>	1024
<b>Max tested</b>	8192
<b>Description</b>	Maximum number of compute nodes that can be allocated to a job. After changing this parameter, Moab must be restarted for changes to take effect. The value cannot exceed that of the <a href="#">MAXNODE on page 994</a> parameter (specified in <code>moab.cfg</code> ). If you specify a value higher than the limit set for the <code>MAXNODE</code> parameter, the value will match <code>MAXNODE</code> . <code>JOBMAXNODECOUNT</code> can also be specified within configure using <code>--with-maxjobsize=&lt;NODECOUNT&gt;</code> .

### MAXGRES

<b>Location</b>	<code>moab.cfg</code> (dynamic parameter)
<b>Default</b>	512

MAXGRES	
<b>Max tested</b>	---
<b>Description</b>	Total number of distinct generic resources that can be managed.

MAXJOB	
<b>Location</b>	moab.cfg (dynamic parameter)
<b>Default</b>	4096
<b>Max tested</b>	500,000
<b>Description</b>	Maximum number of jobs that can be evaluated simultaneously. (Can also be specified within <b>configure</b> using <b>--with-maxjobs=&lt;JOBCOUNT&gt;</b> .)

MAXRSVPERNODE	
<b>Location</b>	moab.cfg (dynamic parameter)
<b>Default</b>	48
<b>Max tested</b>	1024
<b>Description</b>	Maximum number of reservations a node can simultaneously support.

MMAX_ATTR	
<b>Location</b>	moab.h
<b>Default</b>	128
<b>Max tested</b>	512
<b>Description</b>	Total number of distinct node attributes (PBS node attributes/LL node features) that can be tracked.

MMAX_CLASS	
<b>Location</b>	moab.h
<b>Default</b>	24
<b>Max tested</b>	64
<b>Description</b>	Total number of distinct job classes/queues available.

MMAX_FSDEPTH	
<b>Location</b>	moab.h
<b>Default</b>	24
<b>Max tested</b>	32
<b>Description</b>	Number of active fairshare windows.

MAXNODE	
<b>Location</b>	moab.cfg (dynamic parameter)
<b>Default</b>	5120
<b>Max tested</b>	160000
<b>Description</b>	Maximum number of compute nodes supported.

MMAX_PAR	
<b>Location</b>	moab.h
<b>Default</b>	32
<b>Max tested</b>	32
<b>Description</b>	Maximum number of partitions supported.

MMAX_QOS	
<b>Location</b>	moab.h
<b>Default</b>	128
<b>Max tested</b>	128
<b>Description</b>	Total number of distinct QoS objects available to jobs.

MMAX_RACK	
<b>Location</b>	moab.h
<b>Default</b>	200
<b>Max tested</b>	200
<b>Description</b>	Total number of distinct rack objects available within cluster.

MMAX_RANGE	
<b>Location</b>	moab.h
<b>Default</b>	2048
<b>Max tested</b>	2048
<b>Description</b>	<p>Total number of distinct timeframes evaluated.</p> <p><b>Note:</b> This is proportional to the size of the cluster and the number of simultaneously active jobs in the cluster. (Can be specified within <code>./configure</code> using <code>--with-maxrange=&lt;RANGECOUNT&gt;</code>.) Increasing this value will not increase the size of total memory consumed by Moab but may result in minor slowdowns in the evaluation and optimization of reservations.</p>

MMAX_REQ_PER_JOB	
<b>Location</b>	moab.h
<b>Default</b>	5

MMAX_REQ_PER_JOB	
<b>Max tested</b>	64
<b>Description</b>	Total number of unique requirement structures a job can have. Limits the number of <code>-w</code> clauses in the <code>mshow -a</code> command. It also limits the number of <code>-l nodes=X+Y+Z</code> a normal HPC job can have.

JOBMAXTASKCOUNT	
<b>Location</b>	<code>moab.cfg</code> (dynamic parameter)
<b>Default</b>	4096
<b>Max tested</b>	250000
<b>Description</b>	Total number of tasks allowed per job.

Moab currently possesses hooks to allow sites to create local algorithms for handling site specific needs in several areas. The `contrib` directory contains a number of sample local algorithms for various purposes. The `MLocal.c` module incorporates the algorithm of interest into the main code. The following scheduling areas are currently handled via the `MLocal.c` hooks.

- Local Job Attributes
- Local Node Allocation Policies
- Local Job Priorities
- Local Fairness Policies

#### Related topics

- [Appendix I: Considerations for Large Clusters](#)

## Appendix E: Security

Moab provides role and host based authorization, encryption\*, and DES, HMAC, and MD5 based authentication. The following sections describe these features in more detail.

- [Authorization](#)
  - [Role Based Authorization Security Configuration](#)
- [Authentication](#)

- [Mauth Authentication](#)
- [Munge Authentication](#)
- [Server Response Control](#)
- [Interface Development Notes](#)
- [Host Security](#)
  - [Minimal Host Security Enforcement](#)
  - [Medium Host Security Enforcement](#)
  - [Strict Host Security Enforcement](#)
- [Access Portal Security](#)

## Authorization

### Role Based Authorization Security Configuration

Moab provides access control mechanisms to limit how the scheduling environment is managed. The primary means of accomplishing this is through limiting the users and hosts that are trusted and have access to privileged commands and data.

With regard to users, Moab breaks access into three distinct levels.

#### *Level 1 Moab Admin (Administrator Access)*

Level 1 Moab administrators have global access to information and unlimited control over scheduling operations. By default, they are allowed to control scheduler configuration, policies, jobs, reservations, and all scheduling functions. They are also granted access to all available statistics and state information. Level 1 administrators are specified using the [ADMINCFG\[1\]](#) parameter.

#### *Level 2 Moab Admin (Operator Access)*

Level 2 Moab administrators are specified using the [ADMINCFG\[2\]](#) parameter. By default, the users listed under this parameter are allowed to change all job attributes and are granted access to all informational Moab commands.

#### *Level 3 Moab Admin (Help Desk Access)*

Level 3 administrators are specified via the [ADMINCFG\[3\]](#) parameter. By default, they are allowed access to all informational Moab commands. They cannot change scheduler or job attributes.

#### *Configuring Role Based Access*

Moab allows site specific tuning of exactly which functions are available to each administrator level. Moab also provides two additional administrator levels ([ADMINCFG\[4\]](#) and [ADMINCFG\[5\]](#)) that may be used for site specific needs.

To configure Moab role based access, use the [ADMINCFG](#) parameter.

```

ADMINCFG [1]   USERS=root, john SERVICES=ALL NAME=admin
ADMINCFG [3]   USERS=joe, mary  SERVICES=mdiag, mrsvctl, mcredctl NAME=power
ADMINCFG [5]   USERS=joy, blake SERVICES=NONE NAME=users
...
    
```

**i** A *NONE* in services will still allow users to run [showq](#) and [checkjob](#) on their own jobs.

To determine the role of system users and what commands they can run, use the [mcredctl -q role user:<USERID>](#) command.

Using the **SERVICES** attribute of the **ADMINCFG** parameter, access to an arbitrary selection of services can be enabled on a per administrator-level basis. Possible services include the following:

Service	Description
<a href="#">changeparam</a>	Change any scheduling policy or parameter (This command is deprecated. Use <a href="#">mschedctl -m</a> instead).
<a href="#">checkjob</a>	View detailed information for any job.
<a href="#">checknode</a>	View detailed information for any node.
<b>mbal</b>	Perform real-time load-balancing of interactive commands.
<a href="#">mcredctl</a>	View and modify credential attributes.
<a href="#">mdiag</a>	Provide diagnostic reports for resources, workload, and scheduling.
<a href="#">mjobctl</a>	Modify, control, and view jobs.
<a href="#">mnodectl</a>	Modify, control, and view nodes.
<a href="#">mrmctl</a>	Modify, control, and view resource managers.
<a href="#">mrsvctl</a>	Modify, control, and view reservations.
<a href="#">mschedctl</a>	Modify, control, and view scheduler behavior.
<a href="#">mshow</a>	View existing configuration and predicted resource availability.
<a href="#">showstats</a>	View all scheduler and credential statistics.
<a href="#">releaseres</a>	Release all reservations (This command is deprecated. Use <a href="#">mrsvctl -r</a> instead).

Service	Description
<b>runjob</b>	Immediately execute any job (see <a href="#">mjobctl -x</a> ).
<a href="#">setqos</a>	Set QoS on any job (This command is deprecated. Use <a href="#">mjobctl -m</a> instead).
<a href="#">setres</a>	Create any reservation (This command is deprecated. Use <a href="#">mrsvctl -c</a> instead).
<a href="#">setspri</a>	Set system priority on any job (This command is deprecated. Use <a href="#">mjobctl -p</a> instead).
<a href="#">showconfig</a>	Show all scheduler configuration parameters (This command is deprecated. Use <a href="#">mschedctl -l</a> instead).
<a href="#">showres</a>	Show detailed information for any reservation.
<a href="#">showstate</a>	Show detailed information for all jobs, including their locations, and display job error messages, if any.

### Account and Class/Queue Admins

While the **ADMINCFG** parameter allows organizations to provide controlled access to scheduling objects, it does not allow for distribution along organizational boundaries. For example, a site may set up a level 3 administrator to be able to view statistics, diagnose jobs, and modify job priorities; it does not provide a way to differentiate one type of job from another. If a site administrator wanted to allow control based on the queue or account associated with a job, they would best accomplish this using the credential **MANAGERS** attribute.

A credential manager allows a user to be trusted to administer workload and policies for an associated subgroup of jobs. For example, in the configuration below, a number of queue and account managers are configured.

```
CLASSCFG[orion] MANAGERS=johns
CLASSCFG[xray]  MANAGERS=steve2
CLASSCFG[gamma] MANAGERS=steve2,jpw
ACCOUNTCFG[bio] MANAGERS=charles
```

By default, the specified managers can do anything to a job that the actual job owner could do. By default, this would include the ability to view cumulative and per job statistics, see job details, modify job priorities and holds, cancel and preempt jobs, and otherwise adjust policies and constraints within the associated credential.

## Authentication (Interface Security)

Moab supports password-challenge, DES, HMAC, and MD5 based authentication. Authentication protocols may be specified on a per interface basis allowing independent realms of trust with per realm secret keys and even per realm authentication protocols.

## Mauth Authentication

Mauth is a tool provided with Moab that provides client authentication services. With mauth enabled, each client request is packaged with the client ID, a timestamp, and an encrypted key of the entire request generated using the shared secret key.

This tool is enabled by providing a secret key. A random key is selected when the Moab `./configure` script is run and may be regenerated at any time by rerunning `./configure` and rebuilding Moab. If desired, this random key may be overridden by specifying a new key in the protected `.moab.key` file as in the example below:



Moab must be shut down before setting a new secret key. Use the `service moab stop` or `mschedctl -k` commands to shut down Moab.

```
> vi /opt/moab/etc/.moab.key
(insert key)
> cat /opt/moab/etc/.moab.key
XXXXXXXXX
# secure file by setting owner read-only permissions
> chmod 400 /opt/moab/etc/.moab.key
# verify file is owned by root and permissions allow only root to read file
> ls -l /opt/moab/etc/.moab.key
-r----- 1 root root 15 2007-04-05 03:47 /opt/moab/.moab.key
```



All directories in the path containing `.moab.key` must be owned by the root or primary Moab user. It must not be writable by "other" in its permissions.



If `.moab.key` is used, this protected file will need to be on each host that is authorized to run Moab client commands.



By default, this file will be owned by the user root and its contents will be read by the mauth tool which provides client authorization services. If desired, the ownership of this file can be changed so long as this file is readable by the Moab server and the mauth tool. This can be accomplished if the Moab [primary administrator](#), the owner of mauth, and the owner of `.moab.key` are the same.



By default, it is up to the individual cluster administrators to determine whether to use the `.moab.key` file. For sites with source code, the use of `.moab.key` can be mandated by using `./configure --with-keyfile`.



By default, mauth is located in the install `bin` directory. If an alternate name or alternate file location is desired, this can be specified by setting the **AUTHCMD** attribute of the [CLIENTCFG](#) parameter within the `moab.cfg` file as in the following example.

```
CLIENTCFG AUTHCMD=/opt/sbin/mauth
```

### Configuring Peer-Specific Secret Keys

Peer-specific secret keys can be specified using the `CLIENTCFG` parameter. This key information must be kept secret and consequently can only be specified in the `moab-private.cfg` file. With regard to security, there are two key attributes that can be set. (Other resource managers or clients such as Moab Accounting Manager or a SLURM/Wiki interface can also use the attributes to configure their authentication algorithms. The default, unless otherwise stated, is always *DES*. These attributes are listed in the table below:

AUTH	
<b>Format</b>	one of <i>ADMIN1</i> , <i>ADMIN2</i> , or <i>ADMIN3</i>
<b>Default</b>	---
<b>Description</b>	Specifies the level of control/information available to requests coming from this source/peer.
<b>Example</b>	<pre>CLIENTCFG[RM:clusterB] AUTH=admin1 KEY=14335443</pre>

AUTHTYPE	
<b>Format</b>	one of <i>DES</i> , <i>HMAC</i> , <i>HMAC64</i> , or <i>MD5</i> .
<b>Default</b>	<i>DES</i>
<b>Description</b>	Specifies the encryption algorithm to use when generating the message checksum.
<b>Example</b>	<pre>CLIENTCFG[AM:mam] AUTHTYPE=HMAC64</pre>

HOST	
<b>Format</b>	<STRING >
<b>Default</b>	---
<b>Description</b>	Specifies the hostname of the remote peer. Peer requests coming from this host will be authenticated using the specified mechanism. This parameter is optional.
<b>Example</b>	<pre>CLIENTCFG[RM:clusterA] HOST=orx.pb13.com KEY=banana6</pre>

KEY	
<b>Format</b>	<STRING >
<b>Default</b>	---
<b>Description</b>	Specifies the shared secret key to be used to generate the message checksum.
<b>Example</b>	<code>CLIENTCFG[RM:clusterA] KEY=banana6</code>

The **CLIENTCFG** parameter takes a string index indicating which peer service will use the specified attributes. In most cases, this string is simply the defined name of the peer service. However, for the special cases of resource and allocation managers, the peer name should be prepended with the prefix **RM:** or **AM:** respectively, as in `CLIENTCFG[AM:mam]` or `CLIENTCFG[RM:devcluster]`.

**i** The first character of any secret key can be viewed by trusted administrators using specific diagnostic commands to analyze Moab interfaces. If needed, increase the length of the secret keys to maintain the desired security level.

## Munge Authentication

Moab also integrates with MUNGE, an open source authentication service created by Lawrence Livermore National Laboratory (<http://home.gna.org/munge/>). MUNGE works with Moab to authenticate user credentials being passed between the Moab client and the Moab server or from Moab server to Moab server.

To set up MUNGE in a cluster or grid, download and install MUNGE on every node in the cluster or grid by following the installation steps found at <http://home.gna.org/munge/>. The MUNGE secret key must reside on each node in the cluster or grid. Before starting the Moab daemon, the MUNGE daemon must be running on all nodes.

To enable Moab to use MUNGE for authentication purposes, specify the MUNGE executable path in the `moab.cfg` file using **CLIENTCFG** and **AUTHCMD** as in the following example. The MUNGE executable path must reside in each client's `moab.cfg` file as well.

```
CLIENTCFG AUTHCMD=/usr/bin/munge
```

**i** Moab requires that the MUNGE and UNMUNGE executable names be "munge" and "unmunge" respectively. It also assumes that the UNMUNGE executable resides in the same directory as the MUNGE executable.

### Configuring Munge Command Options

Moab also integrates with MUNGE command line options. For example, to set up Moab to use a specific socket that was created when the MUNGE daemon was started, use **CLIENTCFG** and **AUTHCMDOPTIONS** to

specify the newly created socket. The **AUTHCMDOPTIONS** attribute, like **AUTHCMD**, must also reside in the client's `moab.cfg` file.

```
CLIENTCFG      AUTHCMD=/usr/bin/munge
CLIENTCFG      AUTHCMDOPTIONS="-S /var/run/munge/munge.socket.2"
```

## Server Response Control

If a request is received that is corrupt or cannot be authenticated, Moab will report some limited information to the client indicating the source of the failure, such as "bad key," "malformed header," and so forth. In the case of highly secure environments, or to minimize the impact of sniffing or denial of service attacks, Moab can be configured to simply drop invalid requests. This is accomplished by adding the **DROPBADREQUEST** attribute to the **CLIENTCFG** parameter in the `moab-private.cfg` file as in the following example:

```
CLIENTCFG[DEFAULT] DROPBADREQUEST=TRUE
```

## Interface Development Notes

Sample checksum generation algorithm code can be found in the [Socket Protocol Description](#) document.

## Host Security for Compute Resources

Host level security can vary widely from one site to another with everything from pure on-your-honor based clusters to complete encrypted VLAN based network security and government approved per job scrubbing procedures being used. The following documentation describes some best practices in use throughout the industry.

### Minimal Host Security Enforcement

For minimal host security, no additional configuration is required.

### Medium Host Security Enforcement

- Login Access
  - PAM — Enable/disable access by modifying `/etc/security/access.conf`.
- Processes
  - Kill all processes associated with job user (dedicated).
  - Kill all processes associated with job session (dedicated/shared). Use `ps -ju <USER>` or `ps -js <SESSID>`.
- IPC (Inter-Process Communication)
  - Remove shared memory, semaphores, and message queues (use `ipcs/ipcrm`).
  - Remove named pipes.

- Network/Global File System Access
  - Explicitly unmount user home and global file systems.
- Local Temporary File Systems
  - Where possible, mount local file systems read-only.
  - Clear `/tmp`, `/scratch` and other publicly available local file systems.
  - Remove user files with `shred`; `shred` is a Linux command that first overwrites files completely before removing them, preventing remnant data from surviving on the hard drive.

### Strict Host Security Enforcement

- VLAN creation
- Host rebuild
  - U.S Dept. of Energy Disk/File Sanitization ([SCRUB](#))
  - U.S Dept. of Defense Scrubbing [Software](#) (DOD-5520)

### Moab Access Portal Security Overview

The Moab Access Portal (MAP) security model is composed of several different components. First, users will use a Web browser to log in and interact with the Web server running MAP. This communication can be encrypted using industry standard SSL to protect usernames/passwords and other sensitive information that may be accessed by the user. (Instructions on how to set up SSL connections with popular Web servers and servlet engines are readily available on the Internet. A guide for setting up SSL with Apache is available in the [MAP documentation](#).)

When a user logs in via their Web browser, the JSP interface passes this request to a back-end Java infrastructure that then uses an encrypted SSH connection to authenticate the user's credentials at the cluster/grid head node. After the credentials are authenticated and the SSH connection established, further communication between MAP and the cluster/grid head node occurs over the encrypted SSH connection. These three components provide an end-to-end security solution for Web-based job submission and workload management.

## Appendix F: Initial Moab Testing

Moab has been designed with a number of key features that allow testing to occur in a *no risk* environment. These features allow you to safely run Moab in test mode even with another scheduler running whether it be an earlier version of Moab or another scheduler altogether. In test mode, Moab collects real-time job and node information from your resource managers and acts as if it were scheduling live. However, its ability to actually affect jobs (that is, start, modify, cancel, charge, and so forth) is disabled.

Moab offers the following test modes to provide a means for verifying such things as proper configuration and operation:

- [Minimal Configuration Required To Start](#)
  - [Normal Mode](#)
  - [Monitor Mode](#)
  - [Interactive Mode](#)
  - [Simulation Mode](#)

## Scheduler Modes

Central to Moab testing is the **MODE** attribute of the [SCHEDCFG](#) parameter. This parameter attribute allows administrators to determine how Moab will run. The possible values for **MODE** are *NORMAL*, *MONITOR*, *INTERACTIVE*, and *SIMULATION*. For example, to request monitor mode operation, include the following in the `moab.cfg` file:

```
SCHEDCFG MODE=MONITOR
```

### Normal Mode

If initial evaluation is complete or not required, you can place the scheduler directly into *production* by setting the **MODE** attribute of the **SCHEDCFG** parameter to *NORMAL* and (re)starting the scheduler.

### Monitor Mode (or Test Mode)

Monitor mode allows evaluation of new Moab releases, configurations, and policies in a risk-free manner. In monitor mode, the scheduler connects to the resource manager(s) and obtains live resource and workload information. Using the policies specified in the `moab.cfg` file, the monitor-mode Moab behaves identical to a live or normal-mode Moab except the ability to start, cancel, or modify jobs is disabled. In addition, allocation management does not occur in monitor mode. This allows safe diagnosis of the scheduling state and behavior using the various diagnostic client commands. Further, the log output can also be evaluated to see if any unexpected situations have arisen. At any point, the scheduler can be dynamically changed from monitor to normal mode to begin *live* scheduling.

To set up Moab in monitor mode, do the following:

```
> vi moab.cfg
  (change the MODE attribute of the SCHEDCFG parameter from NORMAL to MONITOR)
> moab
```

Remember that Moab running in monitor mode will not interfere with your production scheduler.

### Running Multiple Moab Instances Simultaneously

If running multiple instances of Moab, whether in simulation, normal, or monitor mode, make certain that each instance resides in a different home directory to prevent conflicts with configuration, log, and statistics files. Before starting each additional Moab, set the **MOABHOMEDIR** environment variable in the execution environment to point to the local home directory. Also, each instance of Moab should run using a different [port](#) to avoid conflicts.

**i** If running multiple versions of Moab, not just different Moab modes or configurations, set the `$PATH` variable to point to the appropriate Moab binaries.

To *point* Moab client commands (such as [showq](#)) to the proper Moab server, use the appropriate command line [arguments](#) or set the environment variable `MOABHOMEDIR` in the client execution environment as in the following example:

```
# point moab clients/server to new configuration
> export MOABHOMEDIR=/opt/moab-monitor
# set path to new binaries (optional)
> export PATH=/opt/moab-monitor/bin:/opt/moab-monitor/sbin:$PATH
# start Moab server
> moab
# query Moab server
> showq
```

**i** `moabd` is a safe and recommended method of starting Moab if things are not installed in their default locations.

## Interactive Mode

Interactive mode allows for evaluation of new versions and configurations in a manner different from monitor mode. Instead of disabling all resource and job control functions, Moab sends the desired change request to the screen and asks for permission to complete it. For example, before starting a job, Moab may post something like the following to the screen:

```
Command:  start job 1139.ncsa.edu on node list test013,test017,test018,test021
Accept:   (y/n) [default: n]?
```

The administrator must specifically accept each command request after verifying that it correctly meets desired site policies. Moab will then execute the specified command. This mode is useful in validating scheduler behavior and can be used until configuration is appropriately tuned and all parties are comfortable with the scheduler's performance. In most cases, sites will want to set the scheduling mode to normal after verifying correct behavior.

## Simulation Mode

Simulation mode is of value in performing a *test drive* of the scheduler or when a stable production system exists and an evaluation is desired of how various policies can improve the current performance. See the Simulations documentation for more information.

# Appendix G: Integrating Other Resources with Moab

Moab can interface with most popular resource managers, many cluster services, and numerous general protocols. The following links provide additional information.

## Compute Resource Managers

- TORQUE - [Integration Guide](#), [TORQUE documentation](#)
- SLURM - [Integration Guide](#), <http://www.llnl.gov/linux/slurm>
- WIKI - [WIKI Integration Guide](#)
- Cray XT/TORQUE - Integration Guide ([html](#), [pdf](#)), <http://www.cray.com>

## Provisioning Resource Managers

- xCAT - [Validating an xCAT Installation for Use with Moab](#)

## Hardware Integration

- NUMA - [Integration Guide](#)

## Compute Resource Managers

- [Moab-TORQUE Integration Guide](#) on page 1228
- [Moab-SLURM Integration Guide](#) on page 1232
- [Installation Notes for Moab and TORQUE for Cray](#) on page 1236

### Moab-TORQUE Integration Guide

- [Overview](#)
- [Integration Steps](#)
  - [Install TORQUE](#)
  - [Install Moab](#)
  - [Configure TORQUE](#)
  - [TORQUE/Moab Considerations](#)
- [Current Limitations](#)
- [Troubleshooting](#)

## Install TORQUE

- Install [TORQUE](#)



Keep track of the PBS target directory, *\$PBSTARGDIR*

## Install Moab

- Untar the Moab distribution file.
- Change the directory to the `moab-<version>` directory.
- Run `./configure`.
- Specify the PBS target directory (**\$PBSTARGDIR** from step 2.1) when queried by `./configure`.

Moab interfaces to PBS by utilizing a few PBS libraries and include files. If you have a non-standard PBS installation, you may need to modify `Makefile` and change **PBSIP** and **PBSLP** values and references as necessary for your local site configuration.

The `./configure` script automatically sets up Moab so that the user running `configure` will become the default Primary Moab Administrator (**\$MOABADMIN**). This can be changed by modifying the **ADMINCFG** [1] **USERS**=<USERNAME> line in the Moab configuration file (`moab.cfg`). The primary administrator is the first user listed in the **USERS** attribute and is the ID under which the Moab daemon runs.

Some Tru64 and IRIX systems have a local `libnet` library that conflicts with PBS's `libnet` library. To resolve this, try setting **PBSLIB** to `'${PBSLIBDIR}/libnet.a -lpbs'` in the Moab `Makefile`.

Moab is 64-bit compatible. If PBS/TORQUE is running in 64-bit mode, Moab likewise needs to be built in this manner to use the PBS scheduling API (i.e., for IRIX compilers, add `-64` to **OSCCFLAGS** and **OSLDFLAGS** variables in the `Makefile`).

When starting both TORQUE and Moab it is best to have a small delay between starting the servers. In general (and especially for very fast or very large systems) this is recommended startup procedure:

- Start TORQUE.
- Start Moab with scheduling paused (`moab -P`) to give it a chance to load everything in the checkpoint file and to sync with TORQUE.
- Unpause Moab with `mschedctl -r`.

## General Configuration for All Versions of TORQUE

- Make `$MOABADMIN` a PBS admin.
  - By default, Moab only communicates with the `pbs_server` daemons and the `$MOABADMIN` should be authorized to talk to this daemon (See [suggestions](#) for more information).
- (OPTIONAL) Set default PBS queue, nodecount, and walltime attributes. ( See [suggestions](#) for more information.)
- (OPTIONAL - TORQUE Only) Configure TORQUE to report completed job information by setting the `qmgrkeep_completed` parameter:>

```
> qmgr -c 'set server keep_completed = 300'
```



PBS nodes can be configured as *time shared* or *space shared* according to local needs. In almost all cases, space shared nodes provide the desired behavior.

**i** PBS/TORQUE supports the concept of virtual nodes. Using this feature, Moab can individually schedule processors on SMP nodes. The online [TORQUE](#) documentation describes how to set up the `$PBS_HOME/server_priv/nodes` file to enable this capability. (For example, `<NODENAME> np=<VIRTUAL NODE COUNT>`)

## Version-Specific Configuration for [TORQUE](#)

Do not start the `pbs_sched` daemon. This is the default scheduler for TORQUE; Moab provides this service.

**i** Moab uses PBS's scheduling port to obtain real-time event information from PBS regarding job and node transitions. Leaving the default `qmgr` setting of `set server scheduling=True` allows Moab to receive and process this real-time information.

## Configure Moab

By default, Moab automatically interfaces with TORQUE/PBS when it is installed. Consequently, in most cases, the following steps are not required:

- Specify PBS as the primary resource manager by setting `RMCFG[base] TYPE=PBS` in the Moab configuration file (`moab.cfg`).

If a non-standard PBS installation/configuration is being used, additional Moab parameters may be required to enable the Moab/PBS interface as in the line `RMCFG[base] HOST=$PBSSERVERHOST PORT=$PBSSERVERPORT`. See the [Resource Manager Overview](#) for more information.

**i** Moab's user interface port is set using the `SCHEDCFG` parameter and is used for user-scheduler communication. This port must be different from the PBS scheduler port used for resource manager-scheduler communication.

## TORQUE/Moab Considerations

The default meaning of a node for TORQUE and Moab are not the same. By default, a node is a host in TORQUE. The node may have one or more execution slots (procs) allocated to it in the [\\$TORQUE\\_HOME/server\\_priv/nodes](#) file. However, the number of nodes recognized by TORQUE is equivalent to the number of node entries in the `$TORQUE_HOME/server_priv/nodes` file. A node specification from `qsub` such as `-1 nodes=2:ppn=2` will direct TORQUE to allocate to execution slots on two separate nodes.

Moab is more liberal in its interpretations of a node. To Moab, the `qsub` request above would be interpreted to mean allocate four tasks with at least two tasks on a node. Where TORQUE would require two nodes for the request, Moab will place all four tasks on the name node (host) if four execution slots are available.

If a cluster has four nodes with eight processors each, TORQUE still sees only four nodes. Moab sees 32 nodes. However, if a user made a `qsub` request with `-1 nodes=10`, TORQUE would reject the request because there are only four nodes available. To enable TORQUE to accommodate Moab's more liberal

node interpretation, the server parameter [available\\_resources.nodect](#) can be set as a server parameter in TORQUE. The value of [available\\_resources.nodect](#) should equal at least the number of execution slots in the cluster.

For our example, cluster [available\\_resources.nodect](#) should be `32`. With this parameter set, the user can now make a request such as `-1 nodes=8:ppn=2`. In this example, the user is still limited to a maximum node request of 32.

With [available\\_resources.nodect](#) set in TORQUE, Moab can be directed to honor the default TORQUE behavior by setting [JOBNODEMATCHPOLICY](#) to `EXACTNODE`.

## PBS Features Not Supported by Moab

Moab supports basic scheduling of all PBS node specifications.

## Moab Features Not Supported by PBS

PBS does not support the concept of a job QoS or other extended scheduling features by default. This can be handled using the techniques described in the [PBS Resource Manager Extensions](#) section. See the [Resource Manager Extensions Overview](#) for more information.

## Troubleshooting

On TRU64 systems, the PBS `libpbs` library does not properly export a number of symbols required by Moab. This can be worked around by modifying the Moab `Makefile` to link the PBS `rm.o` object file directly into Moab.

### TORQUE/PBS Integration Guide - RM Access Control

## Server Configuration

Using the PBS [qmgr](#) command, add the Moab administrator as both a *manager* and *operator*.

```
> qmgr
Qmgr: set server managers += <MOABADMIN>@*.<YOURDOMAIN>
Qmgr: set server operators += <MOABADMIN>@*.<YOURDOMAIN>
Qmgr: quit
```

For example:

```
> qmgr
Qmgr: set server managers += staff@*.ucsd.edu
Qmgr: set operators += staff@*.ucsd.edu
Qmgr: quit
```

**i** If desired, the Moab administrator can be enabled as a manager and operator only on the host on which Moab is running by replacing `"*.<YOURDOMAIN>"` with `"<MOABSERVERHOSTNAME>"`.

## Mom Configuration (optional)

If direct Moab to pbs\_mom communication is required, the mom\_priv/config file on each compute node where pbs\_mom runs should be set as in the following example:

```
$restricted *.<YOURDOMAIN>
$clienthost <MOABSERVERHOSTNAME>
```

**i** For security purposes, sites may want to run Moab under a non-root user id. If so, and Moab-pbs\_mom communication is required, the mom\_priv/config files must be world-readable and contain the line '\$restricted \*.<YOURDOMAIN>'. (i.e., '\$restricted \*.uconn.edu')

### TORQUE/PBS Config - Default Queue Settings

## Default Queue

To set the default queue (the queue used by jobs if a queue is not explicitly specified by the user), issue the following:

```
>> qmgr
Qmgr: set system default_queue = <QUEUENAME>
Qmgr: quit
```

## Queue Default Node and Walltime Attributes

To set a default of one node and 15 minutes of walltime for a particular queue, issue the following:

```
> qmgr
Qmgr: set queue <QUEUENAME> resources_default.nodect = 1
Qmgr: set queue <QUEUENAME> resources_default.walltime = 00:15:00
Qmgr: quit
```

## Default System Wide Node and Walltime Attributes

To set system wide defaults, set the following:

```
> qmgr
Qmgr: set server resources_default.nodect = 1
Qmgr: set server resources_default.walltime = 00:15:00
Qmgr: quit
```

### Moab-SLURM Integration Guide

- [Overview](#)
- [SLURM Configuration Steps](#)
- [Moab Configuration Steps](#)
  - [Configuration for Standby and Expedite](#)
  - [Configuration for the Quadrics Switch](#)

- [Authentication](#)
- [Queue/Class Support](#)
- [Policies](#)
- [Moab Queue and RM Emulation](#)
- [SLURM High Availability](#)

## Overview

Moab can be used as the scheduler for the [SLURM](#) resource manager. In this configuration, the SLURM handles the job queue and the compute resources while Moab determines when, where and how jobs should be executed according to current cluster state and site mission objectives.

The documentation below describes how to configure Moab to interface with SLURM.

**i** For Moab-SLURM integration, Moab 6.0 or higher and SLURM 2.2 or higher are recommended. From the [downloads](#) page, the generic version is needed to install SLURM.

## SLURM Configuration Steps

To configure SLURM to utilize Moab as the scheduler, the **SchedulerType** parameters must be set in the `slurm.conf` config file located in the SLURM `etc` directory (`/usr/local/etc` by default)

```
# slurm.conf
SchedulerType=sched/wiki2
```

The **SchedulerType** parameter controls the communication protocol used between Moab and SLURM. This interface can be customized using the [wiki.conf](#) configuration file located in the same directory and further documented in the SLURM [Admin Manual](#).

**Note:** To allow sharing of nodes, the SLURM partition should be configured with 'Shared=yes' attribute.

## Moab Configuration Steps

By default, Moab is built with WIKI interface support (which is used to interface with SLURM) when running the standard `configure` and `make` process.

To configure Moab to use SLURM, the parameter '[RMCFG](#)' should be set to use the **WIKI:SLURM** protocol as in the example below.

```
# moab.cfg
SCHEDCFG[base] MODE=NORMAL
RMCFG[base] TYPE=WIKI:SLURM
...
```

**Note:** The **RMCFG** index (set to *base* in the example above) can be any value chosen by the site. Also, if SLURM is running on a node other than the one on which Moab is running, then the **SERVER** attribute of the [RMCFG](#) parameter should be set.

**Note:** SLURM possesses a [SchedulerPort](#) parameter which is used to communicate with the scheduler. Moab will auto-detect this port and communicate with SLURM automatically with no explicit configuration required. Do NOT set Moab's [SCHEDCFG\[ \] PORT](#) attribute to this value, this port controls Moab client communication and setting it to match the [SchedulerPort](#) value will cause conflicts. With no changes, the default configuration will work fine.

**Note:** If the SLURM client commands/executables are not available on the machine running Moab, SLURM partition and other certain configuration information will not be automatically imported from SLURM, thereby requiring a manual setup of this information in Moab. In addition, the SLURM [VERSION](#) should be set as an attribute on the [RMCFG](#) parameter. If it is not set, the default is version 1.2.0. The following example shows how to set this line if SLURM v1.1.24 is running on a host named Node01 (set using the [SERVER](#) attribute).

```
# moab.cfg with SLURM on Host Node01
RMCFG[base] TYPE=WIKI:SLURM SERVER=Node01 VERSION=10124
...
```

## Configuration for Standby and Expedite Support

SLURM's 'Standby' and 'Expedite' options are mapped to the Moab [QoS](#) feature. By default, when a SLURM interface is detected, Moab will automatically create a 'standby' and an 'expedite' QoS. By default, the 'standby' QoS will be globally accessible to all users and on all nodes and will have a lower than normal priority. Also by default, the 'expedite' QoS will not be accessible by any user, will have no node constraints, and will have a higher than normal priority.

### Authorizing Users to Use 'Expedite'

To allow users to request 'expedite' jobs, the user will need to be added to the 'expedite' QoS. This can be accomplished using the [MEMBERULIST](#) attribute as in the following example:

```
MEMBERULIST
# allow josh, steve, and user c1443 to submit 'expedite' jobs
QOSCFG[expedite] MEMBERULIST=josh,steve,c1443
...
```

### Excluding Nodes for 'Expedite' and 'Standby' Usage

Both 'expedite' and 'standby' jobs can be independently excluded from certain nodes by creating a QoS-based [standing reservation](#).

Specifically, this is accomplished by creating a reservation with a logical-*not* QoS ACL and a hostlist indicating which nodes are to be exempted as in the following example:

```
MEMBERULIST

# block expedite jobs from reserved nodes
SRCFG[expedite-blocker] QOSLIST=!expedite
SRCFG[expedite-blocker] HOSTLIST=c001[3-7],c200
SRCFG[expedite-blocker] PERIOD=INFINITY

# block standby jobs from rack 13
SRCFG[standby-blocker] QOSLIST=!standby
SRCFG[standby-blocker] HOSTLIST=R:r13-[0-13]
SRCFG[standby-blocker] PERIOD=INFINITY
...
```

## Quadrics Integration

If managing a cluster with a Quadrics high speed network, significant performance improvement can be obtained by instructing Moab to allocate contiguous collections of nodes. This can be accomplished by setting the `NODEALLOCATIONPOLICY` parameter to `CONTIGUOUS` as in the example below:

```
# moab.cfg

SCHEDCFG[cluster1]  MODE=NORMAL SERVER=head.cluster1.org
RMCFG[slurm]        TYPE=wiki:slurm
NODEALLOCATIONPOLICY CONTIGUOUS
...
```

## Setting Up Authentication

By default, Moab will not require server authentication. However, if SLURM's `wiki.conf` file (default location is `/usr/local/etc`) contains the `AuthKey` parameter or a secret key is specified via SLURM's `configure` using the `--with-key` option, Moab must be configured to honor this setting. Moab configuration is specified by setting the resource manager `AUTHTYPE` attribute to `CHECKSUM` and the `KEY` value in the `moab-private.cfg` file to the secret key as in the example below.

```
# /usr/local/etc/wiki.conf

AuthKey=4322953
...
```

```
# moab.cfg

RMCFG[slurm]        TYPE=wiki:slurm AUTHTYPE=CHECKSUM
...
```

```
# moab-private.cfg

CLIENTCFG[RM:slurm] KEY=4322953
...
```

**Note:** For the `CHECKSUM` authorization method, the key value specified in the `moab-private.cfg` file must be a decimal, octal, or hexadecimal value, it cannot be an arbitrary non-numeric string.

## Queue/Class Support

While SLURM supports the concept of classes and queues, Moab provides a flexible alternative queue interface system. In most cases, sites can create and manage queues by defining partitions within SLURM. Internally, these SLURM partitions are mapped to Moab [classes](#) which can then be managed and configured using Moab's [CLASSCFG](#) parameter and [mdiag -c](#) command.

## Policies

By default, SLURM systems only allow tasks from a single job to utilize the resources of a compute node. Consequently, when a SLURM interface is detected, Moab will automatically set the [NODEACCESSPOLICY](#) parameter to *SINGLEJOB*. To allow node sharing, the SLURM partition attribute '**Shared**' should be set to *FORCE* in the `slurm.conf` as in the example below:

```
# slurm.conf
PartitionName=batch Nodes=node[1-64] Default=YES MaxTime=INFINITE State=UP
Shared=FORCE
```

## Moab Queue and RM Emulation

With a SLURM system, jobs can be submitted either to SLURM or to Moab. If submitted to SLURM, the standard SLURM job submission language must be used. If jobs are submitted to Moab using the [msub](#) command, then either LSF\*, PBS, or Loadleveler\* job submission syntax can be used. These jobs will be translated by Moab and migrated to SLURM using its native job language.

## SLURM High Availability

If SLURM high availability mode is enabled, Moab will automatically detect the presence of the SLURM BackupController and utilize it if the primary fails. To verify SLURM is properly configured, issue the SLURM command '`scontrol show config | grep Backup`'. To verify Moab properly detects this information, run '`mdiag -R -v | grep FallBack`'.

**Note:** To use SLURM high availability, the SLURM parameter **StateSaveLocation** must point to a shared directory which is readable and writable by both the primary and backup hosts. See the `slurm.conf` man page for additional information.

### Related topics

- [SLURM Admin Manual](#)
- [SLURM's Moab Integration Guide](#)
- [Additional SLURM Documentation](#)
- [Wiki Overview](#)

### Installation Notes for Moab and TORQUE for Cray

## Overview

Moab and TORQUE can be used to manage the batch system for Cray. This document describes how to configure Moab and TORQUE to bring Moab's unmatched scheduling capabilities to the Cray.

New to TORQUE 4.1, TORQUE now handles all communication with ALPS, specifically the pbs\_mom. Previously, communication with ALPS was handled by a combination of Moab, scripts and TORQUE. In the new model, Moab treats TORQUE as a regular TORQUE cluster without any special configuration. TORQUE now uses an extra MOM called the alps\_reporter MOM to communicate with ALPS regarding configured and available resources. From the information reported by the alps\_reporter mom, TORQUE creates a virtual node for each Cray compute node. Previously, TORQUE only reported the login nodes.

**Note:** For clarity this document assumes that your SDB node is mounting a persistent `/var` filesystem from the bootnode. If you have chosen not to use persistent `/var` filesystems please be aware that the instructions below would have to be modified for your situation.

## Upgrade Notes

When upgrading to TORQUE 4.1.0 and using the new Cray model as described in this document, there should be no running jobs. Jobs may be queued but not running.

## Installing TORQUE on a Cray

**i** These instructions are written for a partitioned system, with separate SDB and boot nodes. A combined SDB/boot node configuration is not supported.

For non-partitioned systems, change `sdb-p1` to `sdb` in these instructions.

Before beginning, note the SDB and login nodes' IDs as you will need them throughout the install process.

```
crayadm@smw> ssh root@boot-p1
boot# grep sdb /etc/hosts
10.128.0.32      nid00031      c0-0c0s0n3      sdb001  sdb002
10.131.255.253 sdb sdb-p1 syslog syslog-p1 ufs ufs-p1

boot# grep login /etc/hosts
10.128.0.3      nid00002      c0-0c0s1n0      login  login-p1      login1  castor-p1
```

*In this example, and throughout this page, the login node has NID 2 and the SDB has NID 31.*

### 1. Copy Moab/TORQUE software to SMW and boot node

```
workstation> scp -p /cray/css/release/cray/build/batch/moab-
torque/torque.5.0.0.tar.gz crayadm@smw:/home/crayadm/<yourusername>
workstation> scp -p /cray/css/release/cray/build/batch/moab-torque/moab-8.0.0-
SUSE11-linux-x86_64-torque.tar.gz crayadm@smw:/home/crayadm/<yourusername>

crayadm@smw> cd /home/crayadm/<yourusername>
crayadm@smw> scp -p torque-5.0.0.tar.gz root@boot-p1:/rr/current/software
crayadm@smw> scp -p moab-8.0.0-SUSE11-linux-x86_64-torque.tar.gz root@boot-
p1:/rr/current/software
```

### 2. Install TORQUE. SSH to the boot node and unpack the TORQUE tarball within `xtopview`.

```

crayadm@smw> ssh root@boot-pl
boot# xtopview -m "Installing TORQUE"
default:/# cd /software/
default:/# tar -zxvf torque-5.0.0.tar.gz

(or, if installing on an esMS)
esms# cd /path/to/software/
esms# tar -zxvf torque-5.0.0.tar.gz

```

### 3. Configure, build, and install TORQUE within xtopview.

```

default:/# cd torque-5.0.0
default:/# ./configure --prefix=/opt/torque/5.0.0 --with-server-
home=/var/spool/torque --with-default-server=sdb-p1 --enable-syslog --disable-gcc-
warnings --with-debug --with-modulefiles=/opt/modulefiles --with-job-create
CFLAGS="-DCRAY_MOAB_PASSTHRU"

(or, if installing on an esMS)
esms# cd /path/to/software/torque-5.0.0
esms# ./configure --prefix=/opt/torque/5.0.0 --with-server-home=/var/spool/torque -
-with-default-server=this-esms --enable-syslog --disable-gcc-warnings --with-debug
--with-modulefiles=/cm/local/modulefiles CFLAGS="-DCRAY_MOAB_PASSTHRU"

```

*The server name in the example is **sdb-p1**. Change this to **sdb** on a non-partitioned system.*

### 4. Make and install TORQUE.

```

default:/# make
default:/# make packages
default:/# make install
default:/# ln -sf /opt/torque/5.0.0 /opt/torque/default # The previous default
symlink might need to be deleted first if it exists
default:/# exit

```

### 5. Copy the TORQUE server directory to the Moab server host.

```

boot# cd /rr/current/var/spool
boot# cp -pr torque /snv/31/var/spool
boot# cp -pr torque /snv/2/var/spool

```

*In this example, the SDB node has NID 31, and the login node NID 2.*

### 6. Set up TORQUE on the SDB node.

```

boot# ssh sdb-p1
sdb# export PATH=/opt/torque/default/sbin:/opt/torque/default/bin:$PATH
sdb# cd /software/torque-5.0.0
sdb# ./torque.setup root
root
pbs_server port is: 15001
trqauthd daemonized - port 15005
trqauthd successfully started
initializing TORQUE (admin: root@boot)

You have selected to start pbs_server in create mode.
If the server database exists it will be overwritten.
do you wish to continue y/(n)?

# Type y

sdb# qmgr
Qmgr: set server keep_completed = 60 # Number of seconds to keep completed jobs in
qstat
unset queue batch resources_default.nodes
set server acl_host_enable = true
set server acl_hosts += nid00002
set server acl_hosts += castor-p1 # Where castor-p1 is the hostname of the login
node
set server acl_hosts += sdb-p1
set server submit_hosts += login
set server submit_hosts += login-p1 # Only needed on partitioned systems
set server submit_hosts += castor-p1 # Where castor-p1 is the hostname of the login
node
set server submit_hosts += nid00002 # Where nid00002 is the NID of the login node
set server tcp_timeout = 30
set server query_other_jobs = True
set server disable_server_id_check = True
set queue batch resources_default.mppnppn=16 # On Cascade systems with aprun -j1
set as default
set server cray_enabled = True
set server resources_default.partition = castor # Where "castor" is the same as the
RMCFG[clustername] in moab.cfg
exit

```

#### 7. Get the number of nodes available on the system.

```
sdb# echo Node count is $(( $(apstat -v | grep XT | awk '{print \$3}') ))
```

In the *nodes* / *nodect* commands, put in your system's number of nodes.

```

sdb# qmgr
set server resources_available.nodes = 20
set server resources_available.nodect = 20
set queue batch resources_available.nodes = 20
set queue batch resources_available.nodect = 20
exit

```

#### 8. Create TORQUE nodes file.

```
sdb# vi /var/spool/torque/server_priv/nodes  
  
    castor-p1 alps_login np=1000 # Where "castor-p1" is the hostname of your login  
node  
    sdb-p1 alps_reporter  
  
sdb# exit
```

*The np attribute is the number of processes that can be running at once. This number should be set appropriately high, depending on the number of nodes on the system.*

9. Install the `torque_server` `init.d` script on the SDB node.

**i** There is a known issue with some of the `init` scripts included with TORQUE. If the included `init` script doesn't work for you, use the process below to create a workaround `torque_server` `init.d` script for your system.

```

boot# xtopview -n 31 -m "torque_server init.d"
node/31:/ # touch /etc/init.d/torque_server
node/31:/ # xtspec -n 31 /etc/init.d/torque_server
node/31:/ # chmod a+x /etc/init.d/torque_server
node/31:/ # vi /etc/init.d/torque_server

#!/bin/sh
#
# pbs_server This script will start and stop the PBS Server
#
### BEGIN INIT INFO
# Provides:          pbs_server
# Required-Start:    $local_fs network
# Should-Start:
# Required-Stop:
# Should-Stop:
# Default-Start:    2 3 5
# Default-Stop:
# Description:       Torque is a versatile batch system for SMPs and clusters
### END INIT INFO

PBS_DAEMON=/opt/torque/default/sbin/pbs_server
PBS_HOME=/var/spool/torque
PIDFILE=$PBS_HOME/server_priv/server.lock
export PBS_DAEMON PBS_HOME PIDFILE

# Source the library functions
. /etc/rc.status
rc_reset

[ -f /etc/sysconfig/pbs_server ] && . /etc/sysconfig/pbs_server
[ -x $PBS_DAEMON ] || exit

# How were we called
case "$1" in
  start)
    echo -n "Starting TORQUE Server: "
    ulimit -c unlimited
    if [ -r $PBS_HOME/server_priv/serverdb ]
    then
      startproc $PBS_DAEMON $SERVER_ARGS
    else
      startproc $PBS_DAEMON -t create $DAEMON_ARGS
    fi
    rc_status -v
    ;;
  stop)
    echo -n "Shutting down TORQUE Server: "
    killproc -p $PIDFILE $PBS_DAEMON
    rc_status -v
    ;;
  status)
    echo -n "Checking TORQUE Server: "
    checkproc -p $PIDFILE pbs_server
    rc_status -v
    ;;
  restart)
    $0 stop

```

```

        $0 start
        rc_status
        ;;
    try-restart)
        $0 status >/dev/null && $0 restart
        rc_status
        ;;
    reload|force-reload)
        echo -n "Reloading TORQUE Server: "
        killproc -p $PIDFILE pbs_server -HUP
        rc_status -v
        ;;
    *)
        echo "Usage: torque_server {start|stop|status|try-
restart|restart|force-reload|reload}"
        exit 1
esac
rc_exit

```

10. Install the `torque_mom` `init.d` script on the SDB (or other node that runs the `alps_reporter` service for TORQUE) and login nodes.

**i** There is a known issue with some of the init scripts included with TORQUE. If the included init script doesn't work for you, use the process below to create a workaround `torque_mom` `init.d` script for your system.

*Example 3-177: SDB node*

```

boot# xtopview -n 31 -m "torque_mom init.d"
node/31:/ # touch /etc/init.d/torque_mom
node/31:/ # xtspec -n 31 /etc/init.d/torque_mom
node/31:/ # chmod +x /etc/init.d/torque_mom
node/31:/ # vi /etc/init.d/torque_mom

#!/bin/sh
#
# pbs_mom          This script will start and stop the PBS Mom
#
### BEGIN INIT INFO
# Provides:        pbs_mom
# Required-Start:  $local_fs
# Should-Start:    pbs_server pbs_sched
# Required-Stop:
# Should-Stop:
# Default-Start:   2 3 5
# Default-Stop:
# Description:     Torque is a versatile batch system for SMPs and clusters
### END INIT INFO

PBS_DAEMON=/opt/torque/default/sbin/pbs_mom
PBS_HOME=/var/spool/torque
PIDFILE=$PBS_HOME/mom_priv/mom.lock
export PBS_DAEMON PBS_HOME PIDFILE

ulimit -n 32768
# Source the library functions
. /etc/rc.status
rc_reset

[ -f /etc/sysconfig/pbs_mom ] && . /etc/sysconfig/pbs_mom
[ -x $PBS_DAEMON ] || exit

args=""
if [ -z "$PREVLEVEL" ];then
# being run manually, don't disturb jobs
args="-p"
fi

# How were we called
case "$1" in
  start)
    echo -n "Starting TORQUE Mom: "
    #ulimit -c unlimited
    /sbin/startproc $PBS_DAEMON $args $DAEMON_ARGS
    rc_status -v
    ;;
  purge)
    [ -f /var/lock/subsys/pbs_mom ] && $0 stop
    echo -n "Starting TORQUE Mom with purge: "
    startproc $PBS_DAEMON -r $DAEMON_ARGS
    rc_status -v
    ;;
  stop)
    echo -n "Shutting down TORQUE Mom: "
    /sbin/killproc -p $PIDFILE $PBS_DAEMON
    rc_status -v
    ;;
)

```

```

status)
    echo -n "Checking TORQUE Mom: "
    checkproc -p $PIDFILE $PBS_DAEMON
    rc_status -v
    ;;
restart)
    $0 stop
    sleep 1
    $0 start -p
    rc_status
    ;;
try-restart)
    $0 status >/dev/null && $0 restart
    rc_status
    ;;
reload|force-reload)
    echo -n "Re-reading TORQUE Mom config file: "
    killproc -p $PIDFILE -HUP pbs_mom
    rc_status -v
    ;;
*)
    echo "Usage: torque_mom {start|stop|status|try-restart|restart|force-
reload|reload|purge}"
    exit 1
esac

```

#### Example 3-178: Login nodes

```

boot# xtopview -c login -m "torque_mom init.d"
class/login:/ # touch /etc/init.d/torque_mom
class/login:/ # xtspec -c login /etc/init.d/torque_mom
class/login:/ # chmod +x /etc/init.d/torque_mom
class/login:/ # vi /etc/init.d/torque_mom

# Use the same script as the SDB node above

```

11. Create the MOM configuration file. This must be done on every login node and also the alps\_reporter node (typically the SDB) specified in the TORQUE `server_priv/nodes` file.
  - a. First, determine if the ALPS path needs to be configured in the MOM configuration file.

```

login# which apbasil
/usr/bin/apbasil
# No configuration change needed

login# which apbasil
/opt/cray/alps/5.0.2-2.0500.7827.1.1.ari/bin/apbasil
# MOM configuration change is needed. This path needs to be declared in the mom_
priv/config file.

```

- b. Create and populate the MOM configuration file on the nodes.

```
login and sdb# vi /var/spool/torque/mom_priv/config

$usecp */:/ufs /ufs
$usecp */:/home /home
$usecp */:/home/users /home/users
$usecp */:/scratch /scratch
$usecp */:/lus /lus
$usecp */:/extlus /extlus
$login_node true # For login node
$reporter_mom true # For SDB node
$apbasil_protocol 1.2
$prologalarm 120
$apbasil_path /opt/cray/alps/default/bin/apbasil # Only if needed. Use the path
discovered above.
```

- c. If needed, you can add CPR information to the MOM configuration on the login nodes .

```
$checkpoint_run_exe /opt/cray/blcr/default/bin/cr_run
$checkpoint_script /opt/cray/cprbatchutils/default/libexec/checkpoint.torque
$restart_script /opt/cray/cprbatchutils/default/libexec/restart.torque
$remote_checkpoint_dirs /lus/scratch/BLCR_checkpoint_dir
```

12. Create the `torque.cfg` file (useful if having issues with LDAP users submitting jobs) on the SDB node.

```
sdb# vi /var/spool/torque/torque.cfg

QSUBSENDUID true
VALIDATEPATH FALSE
```

13. Install the `trqauthd` init.d script on the SDB and login nodes.

```
boot# xtopview -n <SDB or login> -m "trqauthd"
node/<SDB or login>:/ # cp /software/torque-5.0.0/contrib/init.d/suse.trqauthd
/etc/init.d/trqauthd
node/<SDB or login>:/ # chmod +x /etc/init.d/trqauthd
node/<SDB or login>:/ # vi /etc/init.d/trqauthd

PBS_DAEMON=/opt/torque/default/sbin/trqauthd
```

14. Start the `trqauthd` daemon on the SDB and login nodes.

```
<SDB or login># /etc/init.d/trqauthd start
```

## Enabling node features for Cray compute nodes

Node features can be set for Cray compute nodes. To add node features to a Cray compute node, use the `cray_compute` keyword on designated nodes in the `nodes` file.:

```
# node_id cray_compute feature_name
2 cray_compute bigmem
```

## Configuring TORQUE for ALPS 1.3

To configure TORQUE for ALPS 1.3, configure the `apbasil_protocol` parameter in `mom_priv/config` and set the `nppcu` server parameter. The `nppcu` parameter has three options that

determine whether to use Hyper-Threading:

Table 3-6: nppcu values

Value	Description
0	Allow ALPS to choose
1	Hyper-Threading disabled (default)
2	Hyper-Threading enabled

When nppcu is set to 0 or 2, pbs\_nodes reports twice as many cores.

### apbasil\_protocol:

```
$apbasil_protocol 1.3
$loglevel 3
```

### nppcu:

```
qmgr -c 'set server nppcu=1'
```

## Installing Moab Workload Manager

1. Unpack the Moab tarball within xtopview.

```
boot# xtopview -m "Installing Moab"
default:/# cd /software/
default:/# tar -zxvf moab-8.0.0-linux-x86_64-torque-xt4.tar.gz
default:/# cd moab-8.0.0
```

2. Configure, build, and install Moab within xtopview.

```
default:/# ./configure --prefix=/opt/moab/8.0.0 --with-homedir=/var/spool/moab --
with-torque=/opt/torque/default --with-modulefiles=/opt/modulefiles --with-xt4

(or, if installing on an esMS)
default:/# ./configure --prefix=/opt/moab/8.0.0 --with-homedir=/var/spool/moab --
with-torque=/opt/torque/default --with-modulefiles=/cm/local/modulefiles --with-xt4

default:/# make install
default:/# ln -sf /opt/moab/8.0.0 /opt/moab/default # The previous default symlink
may need to be deleted first if it exists
default:/# exit
```

3. Configure the moab.cfg file.

```

boot# cd /rr/current/var/spool/moab/etc
boot# vi moab.cfg

    Change the value of SCHEDCFG[Moab] to SERVER=sdb-pl:42559 # Leave the port
    number as whatever default is present
    Change ADMINCFG[1] USERS=root to USERS=root,crayadm # Where "crayadm" is the
    administrative user
    If applicable, change TOOLSDIR from /opt/moab/8.0.0/tools to
    /opt/moab/default/tools

    Change RMCFG[boot]      TYPE=PBS to
    RMCFG[castor]          TYPE=TORQUE # "castor" can be any logical name for the
    partition

    For Moab version 6.x, use TYPE=NATIVE:XT4

    Example:
    RMCFG[tuna]  TYPE=TORQUE SUBMITCMD=/opt/torque/default/bin/qsub
    FLAGS=asyncstart

    Add:
    RMPOLLINTERVAL      00:00:10
    DEFERTIME           00:05:00
    JOBNODEMATCHPOLICY EXACTNODE
    NODECFG[DEFAULT]   OS=linux ARCH=XT
    NODEACCESSPOLICY   SINGLEJOB
    JOBMIGRATEPOLICY   IMMEDIATE
    NODEALLOCATIONPOLICY PRIORITY
    NODECFG[DEFAULT]   PRIORITYF='PRIORITY'
    NODECFG[castor-pl] Partition=login # Use a logical name such as "login" to
    keep the MOM nodes in a separate
    # partition from the compute nodes
    # "castor-pl" in this case is the hostname
    of the login node
    CLIENTCFG[DEFAULT] DEFAULTSUBMITPARTITION=castor # Where "castor" is the name
    of the partition (see RMCFG[castor] above)
    JOBMAXTASKCOUNT   <total number of processors>

    # Comment out USEDATABASE INTERNAL

    # If using a re-purposed compute node as the alps_reporter MOM, add:
    NODECFG[nid00060]  Partition=login # Where nid00060 is the nid of the RCN

    # If necessary to ignore nodes (such as 24 core nodes on a primarily 32 core
    system), add:
    IGNORENODES       57,58 # Where "57" and "58" are node hostnames

```

#### 4. Configure the config.xt4.pl file.

```

boot# vi config.xt4.pl

$basilProtocol = "1.2";

# Uncomment the %loginReplaceTable line, and update the hostnames:
%loginReplaceTable = (nid00002 => "login-p1"); # Where "login-p1" is the exact
hostname of the login node, for example, "castor-p3"

# Uncomment:
$topologyOrdering = 1;

# If on a system that has undergone ALPS standardization (See the apbasil notes
above), replace the $torquePath line with:
my $torquePath = "/opt/torque/default/bin:/usr/bin:/opt/cray/alps/default/bin";

```

#### 5. Copy Moab to the SDB node.

```

boot# cd /rr/current/var/spool/
boot# cp -pr moab /snv/31/var/spool/

For Moab version 6.x only:
boot# mv /snv/31/var/spool/moab/etc/moab.cfg /snv/31/var/spool/moab/

For all versions:
boot# mkdir -p /snv/2/var/spool/moab/etc /snv/2/var/spool/moab/log
boot# cp moab/etc/moab.cfg /snv/2/var/spool/moab/etc/
boot# cp moab/etc/config.xt4.pl /snv/2/var/spool/moab/etc/

```

#### 6. Install the moab init.d script

**i** There is a known issue with some of the init scripts included with Moab. If the included init script doesn't work for you, use the process below to create a workaround moab init.d script for your system.

```

boot# xtopview -n 31 -m "Moab init.d"
node/31:/ # touch /etc/init.d/moab
node/31:/ # xtspec -n 31 /etc/init.d/moab
node/31:/ # chmod a+x /etc/init.d/moab
node/31:/ # vi /etc/init.d/moab
#!/bin/bash
#
# Starts the Moab daemon
#
# chkconfig: 345 96 6
# description: Moab Workload Manager
# processname: moab
#
### BEGIN INIT INFO
# Provides: Moab
# Required-Start: $local_fs $syslog $network $named
# Required-Stop: $local_fs $syslog $network $named
# Default-Start: 3 5
# Default-Stop: 0 1 2 6
# Short-Description: Moab daemon management
# Description: Start Moab Workload Manager
### END INIT INFO
#
# 1. This file should be installed as /etc/init.d/moab
#
# 2. Start Moab with:
#
#     /etc/init.d/moab start
#
# Source function library.
[ -f /etc/rc.status ] || exit 0
. /etc/rc.status

export MOABHOMEDIR=/var/spool/moab
export MOABPARCLEANUP=Full
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/torque/default/lib

prog=moab
path=/opt/moab/default/sbin
RETVAL=0

function start()
{
    echo -n "Starting $prog: "
    ulimit -s unlimited      # Increase stack size to unlimited
    ulimit -c unlimited     # Uncomment to preserve core files

    export MOABNOALLOCMASTER=1
    sleep 2
    startproc $path/moab

    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && touch /var/lock/subsys/moab
    return $RETVAL
}

```

```

function stop()
{
    echo -n "Shutting down $prog: "
    killproc moab
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ] && rm -f /var/lock/subsys/moab
    return $RETVAL
}

function restart()
{
    stop
    sleep 2
    start
}

function condrestart()
{
    if [ -f /var/lock/subsys/capi ] ; then
        restart
    fi
}

function reload()
{
    echo -n $"Reloading $prog: "
    schedctl -R
    RETVAL=$?
    echo
    return $RETVAL
}

case "$1" in
    start)
        start
        rc_status -v
        ;;
    status)
        status moab
        RETVAL=$?
        rc_status -v
        ;;
    stop)
        stop
        rc_status -v
        ;;
    restart)
        restart
        rc_status -v
        ;;
    condrestart)
        condrestart
        ;;
    reload)
        reload
        rc_status -v
        ;;
    *)

```

```

    echo "Usage: $prog {start|stop|restart|reload|status|condrestart}"
    exit 1
esac

exit $RETVAL

```

7. For versions of Moab earlier than 7.1.3 and TORQUE 4.1.4, root must be allowed to submit jobs.

```

boot# ssh sdb
sdb# module load moab torque
sdb# qmgr
qmgr> set server acl_roots = root
qmgr> exit
sdb# /etc/init.d/torque_server restart

sdb# vi /var/spool/moab/etc/moab.cfg # or /var/spool/moab/moab.cfg for Moab version
6

Add
ALLOWROOTJOBS TRUE

sdb# /etc/init.d/moab restart
or
sdb# mschedctl -R

```

8. For versions of Moab newer than 7.1.3, the root user can submit jobs as another user.

```

sdb# qmgr
qmgr> set server managers += root@*
qmgr> exit

sdb# ssh login

login# qsub -I -l mppwidth=1 -P <otherusername>

```

9. Copy your Moab license file to the SDB node in the `/var/spool/moab/` directory.

## Provisioning Resource Managers

- [Validating an xCAT Installation for Use with Moab on page 1252](#)

### Validating an xCAT Installation for Use with Moab

- [Introduction to Validating xCAT Configuration](#)
- [Verifying Node List](#)
- [Reporting Node Status](#)
- [Verifying Hardware Management Configuration](#)
- [Verifying Provisioning Images](#)
- [Verifying VM Migration](#)

## Introduction to Validating xCAT Configuration

This document describes a series of steps to validate xCAT configuration prior to configuring Moab to manage hardware via xCAT. It is assumed the reader is familiar with xCAT and the xCAT configuration on

the target site. This document does not provide xCAT configuration documentation or troubleshooting information; please refer to the [xCAT documentation](#) for such information.

## Verifying Node List

Verify that all nodes that Moab will manage are known to xCAT with the xCAT `nodels` command. Ensure that all expected (and no unexpected) nodes are listed. You may find it useful to create new group names to identify Moab-managed nodes.

```
[root@h0 moab]# nodels hyper,compute
h1
h2
h3
h4
h5
h7
kvmm1
kvmm10
kvmm2
kvmm3
kvmm4
kvmm5
kvmm6
kvmm7
kvmm8
[root@h0 moab]#
```

## Reporting Node Status

Verify that all nodes report their status correctly using the xCAT `nodestat` command. Ensure that all nodes show the correct status (`sshd`, `installing`, `noping`, and so forth); there should not be any timeouts or error messages.

```
[root@h0 moab]# nodestat hyper,compute |sort
h1: pbs,sshd
h2: pbs,sshd
h3: pbs,sshd
h4: pbs,sshd
h5: pbs,sshd
h7: noping
kvmm10: noping
kvmm1: pbs,sshd
kvmm2: pbs,sshd
kvmm3: pbs,sshd
kvmm4: pbs,sshd
kvmm5: pbs,sshd
kvmm6: pbs,sshd
kvmm7: pbs,sshd
kvmm8: noping
kvmm9: noping
[root@h0 moab]#
```

## Verifying Hardware Management Configuration

Verify that all nodes that Moab will manage have hardware management interfaces correctly configured using the xCAT `nodels` and `rpower` commands. After each of the `rpower` commands, verify the requested state was achieved with `rpower stat`.

```
[root@h0 moab]# nodels h1,kvmm1 nodehm.mgt nodehm.power
h1: nodehm.power: ilo
h1: nodehm.mgt: ilo
kvmm1: nodehm.power: kvm
kvmm1: nodehm.mgt: kvm
[root@h0 moab]# rpower h1,kvmm1 off
h1: off
kvmm1: off
[root@h0 moab]# rpower h1,kvmm1 stat
h1: off
kvmm1: off
[root@h0 moab]# rpower h1,kvmm1 boot
h1: on reset
kvmm1: on reset
[root@h0 moab]# rpower h1,kvmm1 stat
h1: on
kvmm1: on
[root@h0 moab]#
```

## Verifying Provisioning Images

Verify that all operating system images that Moab uses are configured correctly in xCAT. For stateful images, test that all combinations of operating system, architecture, and profile install correctly.

```
[root@h0 moab]# rinstall -o centos5.3 -a x86_64 -p hyper h1
h1: install centos3.2-x86_64-hyper
h1: on reset
[root@n100 ~]# sleep 15 && nodestat n05
n05: ping install centos5.3-x86_64-hyper
[root@h0 moab]#
```

For stateless images, test that nodes are able to network boot the images.

```
[root@h0 moab]# nodech h5 nodetype.os=centos5.3 nodetype.arch=x86_64
nodetype.profile=hyper
[root@h0 moab]# nodeset h5 netboot
h5: netboot centos5.3-x86_64-hyper
[root@h0 moab]# rpower h5 boot
h5: on reset
[root@h0 moab]# sleep 60 && nodestat h5
h5: pbs, sshd
[root@h0 moab]#
```

## Verifying VM Migration

If you use VM migration, verify that xCAT can successfully perform migrations using the `rmigrate` command.

```
[root@h0 moab]# rmigrate kvmm7 h1
kvmm7: migrated to h1
[root@h0 moab]# ssh h1 virsh list
Id Name State
-----
33 kvmm1 running
34 kvmm2 running
35 kvmm7 running
```

## Related topics

- [Native Resource Manager Overview](#)
- [Resource Provisioning](#)

## Hardware Integration

- [Moab-NUMA Integration Guide on page 1255](#)

### Moab-NUMA Integration Guide

Scheduling a shared-memory NUMA type system (not the same as a modern SMP-based individual compute node, which cannot share memory between compute nodes) requires some special configuration. Additionally, Moab can use [NODESETs](#) to guarantee feasibility of large memory jobs and to enforce node allocation based on the system's interconnect network topology.

## Configuration

### To integrate Moab and NUMA

1. Configure Moab to schedule large memory jobs. Because Moab creates a partition for each resource manager by default, you must configure the cluster controlled by the resource manager to be a shared-memory system to support jobs spanning multiple nodes/blades. To do so, use the [PARCFG](#) parameter.

```
RMCFG[sys-uv] TYPE=TORQUE
PARCFG[sys-uv] FLAGS=SharedMem
```

*Cluster `sys-uv` is now configured as a shared-memory system to Moab.*

2. Configure **NODESETs** as shown below.

```
NODESETISOPTIONAL FALSE
NODESETATTRIBUTE FEATURE
NODESETPOLICY ONEOF
NODESETPRIORITYTYPE FIRSTFIT
```

*The **NODESET** parameters tell Moab that performing node allocation using node sets is required, that the node set name is a feature name assigned to compute nodes, that a job must fit within the available nodes of one node set, and that Moab should use the first node set that contains sufficient available nodes to satisfy the job's request.*

3. To configure Moab to perform topology-aware node allocation using node sets, you must create a node set definition for each set of nodes that has the same number of maximum network "hops" from any node to every other node within the node set. For an example, see the following sample scenario:

### Use case

The SGI UV 1000 has a two-socket blade with a physical organization of 16 blades within a blade chassis (SGI term is Intra-Rack Unit or IRU), two blade chassis (IRUs) within a rack, and up to four racks within a single UV system. The UV 1000 interconnect network has a topology that requires zero hops between the two sockets on the same physical blade, one hop between an even-odd blade pair (e.g. blades 0 and 1, 2 and 3, etc.), two hops between all even-numbered or all odd-numbered blades within an IRU, three hops maximum between all blades within an IRU, four hops maximum between all even-numbered blades or all odd-numbered blades within a UV system, and five hops maximum between all blades within a UV system.

- a. Define topology-aware node definitions to parallel the compute nodes reachable within a specific hop count. For the UV 1000, this means the sockets of each blade will belong to six separate node set definitions; i.e., one each for 0, 1, 2, 3, 4, and 5 hops).
- b. Define multiple node sets for different nodes reachable in a specific hop count based on the context of where they are in the network topology; that is, you must create a separate and distinct node set definition for each pair of blades reachable with one hop, for each IRU for its nodes reachable in three hops, etc.
- c. Moab node sets are usually defined as compute node features; that is, each node set defined to Moab should appear as a "feature" name on one or more compute nodes. Which node set/feature names appear on each compute node depends on where the compute node is in the interconnect network topology.

Since the SGI UV operating system identifies each blade socket as a separate NUMA node, each NUMA node within a UV system is traditionally an individual compute node to Moab (although TORQUE has the ability to redefine a compute node definition by grouping OS NUMA nodes, which some UV installations do to define a blade as a compute node).

For the sake of illustration, this example assumes each OS NUMA node, which is a UV blade socket, is also a compute node in Moab. This means each compute node (blade socket) will have six feature names assigned, where each feature name must reflect both the compute node's location in the network topology and the hop count the name represents. A feature name is constructed by using the same root name for a hop count and a number for the topology location at the hop-count level.

For example, the root feature name "blade" represents the zero-hop count and the numbers "0", "1", etc, represent the consecutively numbered blades throughout the entire UV system, which yields feature names of "blade0" for the first blade in the system, "blade1" for the second blade, etc, to "blade127" for the last blade in a fully populated 4-rack UV system. To illustrate further, the root feature name "iru" represents the 3-hops count and the numbers "0" through "7" represent the eight IRUs within a full 4-rack UV system.

- d. For each compute node, configure the correct feature name for each of the hop counts possible and its location within the topology at the hop-count level (e.g., blade (0 hops), blade pair (1 hop), odd- or even-numbered nodes within an IRU (2 hops), IRU (3 hops), odd- or even-numbered nodes within the UV (4 hops), and UV system (5 hops)). The following example illustrates the feature names assigned to the compute nodes for an SGI UV 1000 system using the following root feature

names.

- blade (0 hops)
- pair (1 hop)
- eiru (2 hops for even-numbered blades within an IRU)
- oiru (2 hops for odd-numbered blades within an IRU)
- iru (3 hops)
- esys (4 hops for even-numbered blades within a UV system)
- osys (4 hops for odd-numbered blades within a UV system)
- sys (5 hops)

Note that nodes 0 and 1 are not given any feature names. This is because the operating system instance for the UV system runs on the first blade and in order to not adversely affect OS performance, no jobs should run on the same compute resources as the operating system; hence, these nodes have no node set feature names and therefore will never be chosen to run jobs. In addition, some of the first feature names at a specific hop count-level are omitted (such as pair0) since it makes no sense to define them when the first blade is a substantial part of the nodes making up a node set.

The node name of a UV system has the same name as the UV system's host name plus the NUMA node's relative socket number.

```

/var/spool/torque/server_priv/nodes:
sys-uv0
sys-uv1
sys-uv2  blade1      oiru0  iru0  osys  sys
sys-uv3  blade1      oiru0  iru0  osys  sys
sys-uv4  blade2  pair1  eiru0  iru0  esys  sys
sys-uv5  blade2  pair1  eiru0  iru0  esys  sys
sys-uv6  blade3  pair1  oiru0  iru0  osys  sys
sys-uv7  blade3  pair1  oiru0  iru0  osys  sys
sys-uv8  blade4  pair2  eiru0  iru0  esys  sys
sys-uv9  blade4  pair2  eiru0  iru0  esys  sys
sys-uv10 blade5  pair2  oiru0  iru0  osys  sys
sys-uv11 blade5  pair2  oiru0  iru0  osys  sys
sys-uv12 blade6  pair3  eiru0  iru0  esys  sys
sys-uv13 blade6  pair3  eiru0  iru0  esys  sys
sys-uv14 blade7  pair3  oiru0  iru0  osys  sys
sys-uv15 blade7  pair3  oiru0  iru0  osys  sys
sys-uv16 blade8  pair4  eiru0  iru0  esys  sys
sys-uv17 blade8  pair4  eiru0  iru0  esys  sys
sys-uv18 blade9  pair4  oiru0  iru0  osys  sys
sys-uv19 blade9  pair4  oiru0  iru0  osys  sys
sys-uv20 blade10 pair5  eiru0  iru0  esys  sys
sys-uv21 blade10 pair5  eiru0  iru0  esys  sys
sys-uv22 blade11 pair5  oiru0  iru0  osys  sys
sys-uv23 blade11 pair5  oiru0  iru0  osys  sys
sys-uv24 blade12 pair6  eiru0  iru0  esys  sys
sys-uv25 blade12 pair6  eiru0  iru0  esys  sys
sys-uv26 blade13 pair6  oiru0  iru0  osys  sys
sys-uv27 blade13 pair6  oiru0  iru0  osys  sys
sys-uv28 blade14 pair7  eiru0  iru0  esys  sys
sys-uv29 blade14 pair7  eiru0  iru0  esys  sys

```

```

sys-uv30 blade15 pair7 oiru0 iru0 osys sys
sys-uv31 blade15 pair7 oiru0 iru0 osys sys
sys-uv32 blade16 pair8 eiru1 iru1 esys sys
sys-uv33 blade16 pair8 eiru1 iru1 esys sys
sys-uv34 blade17 pair9 oiru1 iru1 osys sys
sys-uv35 blade17 pair9 oiru1 iru1 osys sys
...
sys-uv62 blade31 pair15 oiru1 iru1 osys sys
sys-uv63 blade31 pair15 oiru1 iru1 osys sys
sys-uv64 blade32 pair16 eiru2 iru2 esys sys
sys-uv65 blade32 pair16 eiru2 iru2 esys sys
...
sys-uv126 blade63 pair31 oiru3 iru3 osys sys
sys-uv127 blade63 pair31 oiru3 iru3 osys sys
sys-uv128 blade64 pair32 eiru4 iru4 esys sys
sys-uv129 blade64 pair32 eiru4 iru4 esys sys
...
sys-uv190 blade95 pair47 oiru5 iru5 osys sys
sys-uv191 blade95 pair47 oiru5 iru5 osys sys
sys-uv192 blade96 pair48 eiru6 iru6 esys sys
sys-uv193 blade96 pair48 eiru6 iru6 esys sys
...
sys-uv252 blade126 pair63 eiru7 iru7 esys sys
sys-uv253 blade126 pair63 eiru7 iru7 esys sys
sys-uv254 blade127 pair63 oiru7 iru7 osys sys
sys-uv255 blade127 pair63 oiru7 iru7 osys sys

```

- Define the order in which Moab should check node sets for available nodes. Since the `NODESETPRIORITYTYPE` has a value of `FIRSTFIT`, the node sets must be ordered from smallest to largest so Moab will always choose the node set with the fewest nodes required to satisfy the job's request. This means listing all blades, blade pairs, even and odd IRUs, IRUs, even and odd system, and system, respectively.

```

moab.cfg:
NODESETLIST
blade1,blade2,blade3,...,blade127,pair1,pair2,pair3,...,pair63,eiru0,oiru0,eiru1,oiru1,
...,eiru7,oiru7,iru0,iru1,...,iru7,esys,osys,sys

```

- Configure Moab to use the `PRIORITY NODEALLOCATIONPOLICY`. This allocation policy causes Moab to allocate enough nodes to fulfill a job's processor and memory requirement.

```

NODEALLOCATIONPOLICY PRIORITY

```

- Set `NODEACCESSPOLICY` to `SINGLEJOB` to ensure that Moab will schedule large memory requests correctly and efficiently. This is necessary even when a job uses only the memory of a NUMA node.

```

NODEACCESSPOLICY SINGLEJOB

```

The policy `SINGLEJOB` tells Moab not to allow jobs to share NUMA resources (cores and memory), which for a shared-memory system is very important for fast job execution. For example, if Moab scheduled a job to use the cores of a NUMA node where memory is used by another job, both jobs would execute slowly (up to 10 times more slowly).

## Job Submission

Jobs can request processors and memory using the `-l nodes=<number of cpus>` and `-l mem=<amount of memory>` syntaxes. You should not have `JOBNODEMATCHPOLICY EXACTNODE` configured on a NUMA system. You must use the `sharedmem` job flag on submission to force the job to

run only on a sharedmem partition or cluster and to indicate that the job can span multiple nodes. For example:

```
qsub -l nodes=3,mem=640sgb,flags=sharedmem
```

## Appendix H: Interfacing with Moab (APIs)

Moab provides numerous interfaces allowing it to monitor and manage most services and resources. It also possesses flexible interfaces to allow it to interact with peer services and applications as both a broker and an information service. This appendix is designed to provide a general overview and links to more detailed interface documentation.

- [H.1 Moab Query and Control APIs](#)
  - Allow external portals and services to obtain information about compute resources, workload, and usage statistics.
- [H.2 Resource Management Interfaces](#)
  - Allow Moab to monitor, schedule, and control services and resources.
- [H.3 Identity and Credential Management Interfaces](#)
  - Allow monitoring and active management of user configuration, credentials, policies, and usage information.
- [H.4 Accounting and Event Interfaces](#)
  - Allow import/export of accounting and event information to external entities.
- [H.5 Job Submission and Management Interface](#)
  - Query resource availability, submit, modify, and manage jobs, and query the status of active and completed jobs.
- [H.6 Grid Services API](#)
  - Provide and use information, data, job, and resource management services in a distributed environment.

Moab interfaces to systems providing various services and using various protocols. This appendix is designed to assist users who want to enable Moab in new environments using one of the existing interfaces. It does not cover the steps required to create a new interface.

### H.1 Query and Control APIs

The Moab Cluster and Grid Suites provides a (Moab) workload manager server that supports a broad array of client services. These services can be directly accessed via Moab client commands.

## H.1.1 CLI (Command Line Interface) XML API

All Moab client commands can report results in XML format to allow the information to be easily integrated into peer services, portals, databases, and other applications. To request that a client command report its output in XML, specify the `--format=xml` flag as in the following example:

```
> showq --format=xml
<Data>
<Object>queue</Object>
<cluster LocalActiveNodes="1" LocalAllocProcs="1" LocalIdleNodes="0"
LocalIdleProcs="3" LocalUpNodes="1"
LocalUpProcs="4" RemoteActiveNodes="0" RemoteAllocProcs="0" RemoteIdleNodes="0"
RemoteIdleProcs="0"
RemoteUpNodes="0" RemoteUpProcs="0" time="1128451812"></cluster>
<queue count="1" option="active">
<job AWDuration="11672" EEDuration="1128451812" Group="[DEFAULT]" JobID="Moab.2"
MasterHost="cw2" PAL="2"
QOS="bug3" ReqAWDuration="54000" ReqNodes="1" ReqProcs="1" RsvStartTime="1128451812"
RunPriority="0"
StartPriority="1" StartTime="1128451812" StatPSDed="11886.580000"
StatPSUtl="11886.580000" State="Running"
SubmissionTime="1128451812" SuspendDuration="0" User="smith"></job>
</queue>
<queue count="1" option="eligible">
<job EEDuration="1128451812" Group="jacksond" JobID="customer.35" QOS="bug"
ReqAWDuration="3600"
ReqProcs="1" StartPriority="1" StartTime="0" State="Idle"
SubmissionTime="1128451812" SuspendDuration="0"
User="johnson"></job>
<queue><queue count="0" option="blocked"></queue>
</Data>
```

### Common Query/Control Services

- jobs
  - query status - [mdiag -j](#) ([XML details](#))
  - submit - [msub](#) ([XML format](#))
  - cancel - [mjobctl -c](#)
- nodes
  - query status - [mdiag -n](#) ([XML details](#))
  - create resource reservation - [mrsvctl -c](#)
  - destroy resource reservation - [mrsvctl -r](#)

## H.2 Resource Management Interfaces

Moab can monitor, schedule, and control services and resources using multiple protocols. These protocols include the following:

- [LDAP](#)
- [script/flat file](#)
- [Resource Manager Specific Interfaces](#) - LSF, SGE, TORQUE, PBSPro, Loadleveler, and so forth

Using the resource manager interfaces, Moab can do the following:

- monitor resources (compute host, network, storage, and software license based resources)
  - load configuration, architecture, and feature information
  - load state, utilization, and workload information
  - load policy and ownership information
- manage resources
  - dynamically reconfigure and reprovision resource hardware (processors, memory, etc.)
  - dynamically reconfigure and reprovision resource software (operating system, application software, filesystem mounts, etc.)
  - dynamically reconfigure and reprovision resource security (VPN's, VLAN's, host security, etc.)
- monitor workload (batch jobs, interactive jobs, persistent services, dynamic services, distributed services)
  - load state, resource requirement, and required environment information
  - load user, group, and credential information
  - load utilization, resource allocation, and policy information
- manage workload
  - migrate jobs from one resource to another (intra-cluster and inter-cluster)
  - modify jobs for translation and optimization purposes
  - suspend, resume, checkpoint, restart, and cancel jobs
- query cluster policies and configuration

### H.3 Identity and Credential Management Interfaces

Moab's identity and credential management interfaces allow Moab to exchange credential and user configuration, access, policy, and usage information.

- [Identity Manager](#)
- Allocation Manager
- [Moab Workload Manager for Grids](#)

### H.4 Accounting Interfaces

Moab accounting interfaces allow Moab to export local utilization statistics, events, and accounting information to site specific scripts.

- [Accounting Interface](#)

## H.6 Job Submission and Management Interface

Moab provides interfaces to enable the following services:

- Resource Availability Query
  - Determine quantity, state, and configuration of configured resources (idle, busy, and down nodes)
  - Determine quantity and configuration of all available resources (idle nodes)
  - Determine resources available subject now and in the future for potential job
  - Determine best target cluster destination for potential job
  - Determine largest/longest job which could start immediately
  - Determine estimated start time for potential job
  - Determine earliest guaranteed start time for potential job
- Reserve Resources
  - Reserve specific resources for desired time frame
- Submit Job ([XML format](#))
  - Submit job to specific cluster
  - Submit job to global job queue
- Manage Job
  - Hold job
  - Adjust job priority
  - Modify job executable, args, data requirements, job dependencies, duration, hostcount, or other attributes
  - Suspend/resume job
  - Checkpoint/requeue job
  - Cancel job
  - Migrate job
  - Adjust job quality of service (QoS)
- Query Job
  - Determine job state, utilization, or output results for idle, active, or completed job
  - Determine estimated start time
  - Determine guaranteed start time

## H.7 Grid Interfaces

Moab provides interfaces to allow interaction with various grid brokers and services. These interfaces allow Moab to provide services as well as utilize services.

### Services Utilized

- Information Services (import and utilize information service data in making scheduling decisions)
- Job Migration
- Data Migration
- Credential Mapping
- Security and Delegation

See [Moab Workload Manager for Grids](#) for more information on utilized services.

### Services Provided

- Information Services (provide resource, workload, and credential information)
- Job Migration
- Data Migration
- Credential Mapping

See [Moab Workload Manager Grid Basics](#) for more information on provided services.

## Appendix I: Considerations for Large Clusters

- [I.1 Resource Manager Scaling](#)
- [I.2 Handling Large Numbers of Jobs](#)
- [I.3 Handling Large Numbers of Nodes](#)
- [I.4 Handling Large Jobs](#)
- [I.5 Handling Large SMP Systems](#)
- [I.6 Server Sizing](#)

There are several key considerations in getting a batch system to scale.

### I.1 Resource Manager Scaling

#### Proper Resource Manager Configuration

- [TORQUE](#)
  - [General Scaling Overview](#)

- OpenPBS/PBSPro
  - Manage Direct Node Communication with [NODEPOLLFREQUENCY](#)

## I.2 Handling Large Numbers of Jobs

### Set a minimum RMPOLLINTERVAL

With event driven resource managers like TORQUE, each time a job is submitted the resource manager notifies the scheduler. In an attempt to minimize response time, the scheduler starts a new scheduling cycle to determine if the newly submitted job can run. In systems with large numbers of jobs submitted at once, this might not result in the desired behavior for two reasons. First, by scheduling at every job submission Moab schedules newly submitted jobs onto available resources in a first come, first served basis rather than evaluating the entire group of new jobs at once and optimizing the placement accordingly. Second, by launching a scheduling iteration for every job submitted, Moab places a heavy load on the resource manager. For example, if a user were to submit 1000 new jobs simultaneously, for each job submitted, the resource manager contacts the scheduler, the scheduler starts a new iteration, and in this iteration, the scheduler contacts the resource manager requesting updated information on all jobs and resources available.

Setting a minimum **RMPOLLINTERVAL** causes the scheduler to not process jobs as quickly as they are submitted, but rather to wait a minimum amount of time to allow more jobs be submitted and to process these new jobs in groups.

```
RMPOLLINTERVAL 30,60
```

*If the system is busy, schedule every 30 seconds. If it is not busy, schedule every 60 seconds.*

### Reduce command processing time

If your system's scheduling cycle regularly takes longer than the [CLIENTTIMEOUT](#) value, you can configure Moab to fork a copy of itself that will respond to certain information-only client commands ([checkjob](#), [showbf](#), [showres](#), and [showstart](#)). This enables you to run intense diagnostic commands while Moab is in the middle of its scheduling process.

When you set **UIMANAGEMENTPOLICY FORK**, Moab forks a copy of itself that will listen for client commands on a separate port, which you must configure with **CLIENTUIPORT**. This forked process responds to `checkjob`, `showbf`, `showres`, and `showstart` until the main scheduling cycle has finished. After that, it is killed and the normal process resumes responding to client commands. Moab prints a disclaimer at the top of each command that was populated by the forked process stating that the information may be a few seconds stale.

*Example 3-179: Sample configuration*

```
UIMANAGEMENTPOLICY    FORK
CLIENTUIPORT          41560
```

*Moab forks a copy of itself on port 41560, where it will watch for checkjob, showbf, showres, and showstart commands until the main scheduling process completes.*

**Example 3-180: Sample command output**

```

$ checkjob 34
-----
NOTE: The following information has been cached by the remote server
and may be slightly out of date.
-----

job 34

State: Idle
Creds: user:wightman group:company class:batch
WallTime: 00:00:00 of 00:01:00
SubmitTime: Thu May 22 14:17:06
(Time Queued Total: 00:00:18 Eligible: 00:00:18)

TemplateSets: DEFAULT
Total Requested Tasks: 1

Req[0] TaskCount: 1 Partition: ALL

SystemID: scale
SystemJID: 34

IWD: $HOME/test/scale
SubmitDir: $HOME/test/scale
Executable: sleep 60

```

**Limited Job Checkpointing**

Use the [LIMITEDJOBBCP on page 987](#) parameter. By default, Moab will checkpoint information about every job it reads from its resource managers. When a cluster routinely runs more than 15000 jobs, they may see some speed-ups by limiting which jobs are checkpointed. When **LIMITEDJOBBCP** is set to **TRUE**, Moab will only checkpoint jobs that have a hold, a system priority, jobs that have had their QoS modified, and a few other limited attributes. Some minimal statistical information is lost for jobs that are not checkpointed.

**Minimize Job Processing Time**

Use the [ENABLEHIGHTHROUGHPUT on page 950](#) parameter. By default, Moab processes all job attributes, filters, remap classes, job arrays, and other information when a job is submitted. This requires full access to the Moab configuration and significantly increases the processing time Moab needs when jobs are submitted. By setting **ENABLEHIGHTHROUGHPUT** to **TRUE**, Moab stores the job information in an internal queue and returns the job ID immediately. The internal queue is processed when Moab begins its next scheduling iteration. This enables Moab to process hundreds of jobs per second rather than 20-30 per second. Because the jobs are processed in a separate queue after the job has been returned, it is recommended that [MAILPROGRAM](#) be configured. Moab will send an email to the user if a job is rejected.

Because the job is not fully processed, some attributes may change after the job has been submitted. For example, when a job class is remapped, the new class is not reflected until Moab begins its next scheduling iteration. Additionally, job arrays are not instantiated until Moab begins its next scheduling cycle.

**i** If **ENABLEHIGHTHROUGHPUT** on page 950 is **TRUE**, you must set **NODEALLOCATIONPOLICY** on page 999 to **FIRSTAVAILABLE**.

## Load all Non-Completed Jobs at Startup

Use the **LOADALLJOB** on page 988 parameter. By default, Moab loads non-complete jobs for active resource managers only. By setting **LOADALLJOB** to **TRUE**, Moab will load all non-complete jobs from all checkpoint files at startup, regardless of whether their corresponding resource manager is active.

## Reducing Job Start Time

Use the **ASYNCSTART** parameter. By default, Moab will launch one job at a time and verify that each job successfully started before launching a subsequent job. For organizations with large numbers of very short jobs (less than 2 minutes in duration), the delay associated with confirming successful job start can lead to productivity losses. If tens or hundreds of jobs must be started per minute, and especially if the workload is composed primarily of serial jobs, then the resource manager **ASYNCSTART** flag may be set. When set, Moab will launch jobs optimistically and confirm success or failure of the job start on the subsequent scheduling iteration. Also consider adding the **ASYNCDELETE** flag if users frequently cancel jobs.

## Reducing Job Reservation Creation Time

Use the **RMCFG** on page 1035 **JOBRSVRECREATE** on page 598 attribute. By default, Moab destroys and re-creates job reservations each time a resource manager updates any aspect of a job. Historically, this stems from the fact that certain resource managers would inadvertently or intentionally migrate job tasks from originally requested nodes to other nodes. To maintain synchronization, Moab would re-create reservations each iteration thus incorporating these changes. On most modern resource managers, these changes never occur, but the effort required to handle this case grows with the size of the cluster and the size of the queue. Consequently, on very large systems with thousands of nodes and thousands of jobs, a noticeable delay is present. By setting **JOBRSVRECREATE** to **FALSE** on resource managers that do not exhibit this behavior, significant time savings per iteration can be obtained.

## Optimizing Backfill Time

Use the **OPTIMIZEDBACKFILL** flag. Speeds up backfill when a system reservation is in use.

## Constraining Moab Logging - LOGLEVEL

Use the **LOGLEVEL** on page 990 parameter. When running on large systems, setting **LOGLEVEL** to 0 or 1 is normal and recommended. Only increase **LOGLEVEL** above 0 or 1 if you have been instructed to do so by Moab support.

## Preemption

When preemption is enabled Moab can take considerably more time scheduling jobs for every scheduling iteration. Preemption increases the number of options available to Moab and therefore takes more time for Moab to optimally place jobs. If you are running a large cluster or have more than the usual amount

of jobs (>10000), consider disabling preemption. If disabling preemption is not possible, consider limiting its scope to only a small subset of jobs (as both preemptors and preemptees).

### Handling Transient Resource Manager Failures

Use the **RMCFG** [MAXITERATIONFAILURECOUNT](#) on page 600 attribute.

### Constrain the number of jobs preempted per iteration

Use the [JOBMAXPREEMPTPERITERATION](#) parameter.

**i** For very large job count systems, configuration options controlling the maximum supported limits may need to be adjusted including the maximum number of [reservations](#) and the maximum number of supported evaluation [ranges](#).

### Scheduler settings

If using Moab, there are a number of parameters which can be set on the scheduler which may improve TORQUE performance. In an environment containing a large number of short-running jobs, the **JOBAGGREGATIONTIME** parameter can be set to reduce the number of workload and resource queries performed by the scheduler when an event based interface is enabled. Setting **JOBAGGREGATIONTIME** instructs the scheduler to ignore events coming from the resource manager and to scheduling at regular intervals, rather than around resource manager events. If the `pbs_server` daemon is heavily loaded and PBS API timeout errors (i.e. "Premature end of message") are reported within the scheduler, the **TIMEOUT** attribute of the **RMCFG** parameter may be set with a value of between 30 and 90 seconds.

## 1.3 Handling Large Numbers of Nodes

For very large clusters (>= 10,000 processors) default scheduling behavior may not scale as desired. To address this, the following parameters should be considered:

Parameter	Recommended Settings
<a href="#">RMPOLLINTERVAL</a>	In large node environments with large and long jobs, scheduling overhead can be minimized by increasing <b>RMPOLLINTERVAL</b> above its default setting. If an event-driven resource management interface is available, values of two minutes or higher may be used. Scheduling overhead can be determined by looking at the scheduling load reported by <a href="#">mdiag -S</a> .
<a href="#">LIMITEDNODECP</a>	Startup/shutdown time can be minimized by disabling full node state checkpointing that includes some statistics covering node availability.
<b>SCHEDCFG</b> <b>FLAGS="</b> <a href="#">FASTRSVSTARTUP</a> <b>on page 1391</b>	When you have reservations on a large number of nodes, it can take Moab a long time to recreate them on startup. Setting the <b>FASTRSVSTARTUP</b> scheduler flag greatly reduces startup time.

\* For clusters where the number of nodes or processors exceeds 50,000, the maximum stack size for the shell in which Moab is started may need to be increased (as Moab may crash if the stack size is too small). On most Unix/Linux based systems, the command `ulimit -s unlimited` may be used to increase the stack size limit before starting Moab. This may be placed in your Moab startup script.

 See [Appendix D](#) for further information on default and supported object limits.

Avoid adding large numbers of [NODECFG](#) lines in the `moab.cfg` or `moab.d/*.cfg` files to keep the Moab boot time low.

For example, adding a configuration line to define features for each node in a large cluster (such as `NODECFG[x] Features+=green,purple`) can greatly increase the Moab boot time. If Moab processes 15 node configuration lines per second for a 50,000-node system, it could add approximately 55 minutes of node configuration processing to the Moab boot time.

In this case, it is better to define the node features in the resource manager configuration.

## 1.4 Handling Large Jobs

For large jobs, additional parameters beyond those specified for [large node](#) systems may be required. These include settings for the maximum number of [tasks per job](#), and the maximum number of [nodes per job](#).

## 1.5 Handling Large SMP Systems

For large-way SMP systems (> 512 processors/node) Moab defaults may need adjustment.

Parameter	Recommended Settings
<a href="#">MAXRSVPERNODE</a>	By default, Moab does not expect more than 48 jobs per node to be running or have future reservations. Increasing this parameter to a value larger than the expected maximum number of jobs per node is advised.

## 1.6 Server Sizing

See Hardware and Software Requirements for recommendations.

Related topics

- [Appendix D](#): Adjusting Default Limits

## Appendix J: Configuring Moab as a Service

Scripts that follow can be used to start up Moab services automatically upon a reboot. To enable a service script, copy the script to `/etc/rc.d/init.d/S97moab`, edit the file to make needed localization changes (adjust binary paths, execution user, etc), and add links to the `rc3.d` and `rc5.d` directories as in the example that follows:

```

> cp mwm.service /etc/rc.d/init.d/S97moab
> vi /etc/rc.d/init.d/S97moab
   (make needed localizations)
> ln -s /etc/rc.d/init.d/S97moab /etc/rc.d/rc3.d
> ln -s /etc/rc.d/init.d/S97moab /etc/rc.d/rc5.d

```

## J.1 Moab Workload Manager Service Scripts

- [Moab Workload Manager Script](#)
- [Moab Workload Manager + TORQUE Script](#)

## J.2 Moab Grid Scheduler Service Script

- [sample script](#)

# Appendix K: Migrating from 3.2

## Overview

This guide is intended to help facilitate migrating from Maui to Moab. If you do not have Moab yet, you can download a [free evaluation version](#). At a high level, migrating from Maui 3.2 to Moab involves minimal effort. In fact, Moab fully supports all Maui parameters and commands. Migration can consist of nothing more than renaming `maui.cfg` to `moab.cfg` and launching Moab using the `moab` command. With this migration, the biggest single issue is becoming aware of all the new facilities and capabilities available within Moab. Beyond this, migration consists of a few minor issues that may require attention such as some [statistics and priorities](#).

Another approach of migrating from Maui to Moab is to configure Moab in Monitor mode and run it beside Maui. Maui will continue to perform the scheduling and control workload. Moab will simply monitor the cluster environment using the policies configured in `moab.cfg`. Moab will not have the ability to affect workload, providing a safe and risk-free environment to evaluate Moab without affecting your production environment. You can also have Moab capture resource and workload trace files and allow Moab to simulate what it would have done if it controlled workload. When you feel comfortable with and want to run Moab live on your cluster, all you need to do is change the mode to NORMAL, stop Maui, and restart Moab. Current jobs will remain running and Moab will take over control of scheduling.

As with any migration, we suggest that you back up important files such as the following: `maui.cfg`, `maui.log` and `maui.ck`.

[View the Flash demo of migrating from Maui to Moab.](#)

## Migrating from Maui to Moab

1. Install Moab Workload Manager. (Installation Instructions)
2. Copy your `maui.cfg` file to the `MOABHOMEDIR/etc (/opt/moab/etc)` and rename it `moab.cfg`.
3. Stop Maui.
4. Start Moab.

5. If Applicable: Re-apply those configurations found in the [Statistics and Checkpointing](#) section that need adjustment after migration as well as any parameters in `moab.cfg` that point to a Maui file like `maui.log`.

## Running Maui and Moab Side-By-Side

1. Install Moab Workload Manager on your cluster. (Installation steps will differ slightly from a typical installation.)
  - a. Run `./configure`.
  - b. Run `make`.
  - c. You will need to set your `MOABHOMEDIR` environment variable to the location where you built Moab by typing `export MOABHOMEDIR=[make directory]`.
2. To have Moab use all the same policies as Maui, copy `maui.cfg` to the `MOABHOMEDIR/etc` and rename it `moab.cfg`.
  - You can also start your `moab.cfg` file from scratch. Just use the `moab.cfg` already in the `MOABHOMEDIR/etc`.
3. Make sure that the port in `moab.cfg` is different than the port used in `maui.cfg`.
4. In the `moab.cfg` file, add the parameter, `SERVERMODE=MONITOR`.
  - If you used the `moab.cfg` from scratch, on the **SCHEDCFG** line add `MODE=MONITOR`.
5. You will need to either put the Moab commands in your environment path (located in `MOABHOMEDIR/bin`) or run the commands from their location if you still want to use the Maui commands in your environment path.
6. Run Moab Workload Manager using the `moab` command located in `MOABHOMEDIR/bin`.

## Other Notes

The following are minor differences between Maui and Moab and changes you may need to make:

## File Naming

Moab uses slightly different naming than Maui. The following table displays these changes:

File	Maui	Moab
<b>executable</b>	<code>maui</code>	<code>moab</code>
<b>logs</b>	<code>maui.log</code>	<code>moab.log</code>
<b>configuration file</b>	<code>maui.cfg</code>	<code>moab.cfg</code>

## Statistics and Checkpointing

Moab supports Maui version 3.2 or higher workload traces (statistics) allowing it to process historical statistics based on these traces as well as generate simulations based on them. No changes are required to use these statistics. See the Simulation Configuration documentation for more information on trace files. You can also view a [flash demonstration](#) of the simulation mode.

Moab does not support the Maui 3.2 checkpointing format. Because of this, state information checkpointed under Maui will not be available at the time of the migration. The loss of this information will have the following impact:

- Admin reservations, if any, will need to be re-created.
- Processed credential and scheduler statistics (displayed by `showstats`) will be lost.
- Admin job system priority configured by the `setspri` command and QoS assignments configured by the `setqos` command, if any, will be lost.

## Verify Configuration File Compatibility

The command `mdiag -C` will perform diagnostics on your new configuration file and may prove helpful in identifying any issues.

## Environment Variables

Scheduler environment variables are supported under Moab with obvious naming changes. Sample environment variables follow:

Maui	Moab
<code>MAUIHOMEDIR</code>	<code>MOABHOMEDIR</code>
<code>MAUIDEBUG</code>	<code>MOABDEBUG</code>
<code>MAUICRASHVARIBALE</code>	<code>MOABCRASHVARIABLE</code>
<code>MAUIENABLELOGBUFFERING</code>	<code>MOABENABLELOGBUFFERING</code>
<code>MAUIRECOVERYACTION</code>	<code>MOABRECOVERYACTION</code>
<code>MAUI-COMMANDS-PATH</code>	<code>MOAB-COMMANDS-PATH</code>
<code>MAUIENABLELOGBUFFERING</code>	<code>MOABENABLELOGBUFFERING</code>

## Appendix R: Node Allocation Plug-in Developer Kit

- [R.1 Overview](#)
  - [R.1.1 Writing the plugin](#)
    - [R.1.1.1 API & Data Structures](#)
  - [R.1.2 Moab configuration](#)
    - [R.1.2.1 Moab.cfg](#)
    - [R.1.2.2 Syntax rules](#)
    - [R.1.2.3 Troubleshooting](#)

### R.1 Overview

Each time Moab schedules a job, it must choose the nodes on which the job will run. Moab uses the Node Allocation policy to select the available nodes to be used. Because there are so many different systems and cluster topologies, you now have the ability to create and use a node allocation plugin for allocating nodes based on your cluster's interconnect topology.

The plugin policy allows you to write your own algorithm to choose which nodes will be used. This algorithm is contained in a shared library that Moab loads at run time.

To obtain the Plug-in Developer Kit (PDK) with the header file and example code, contact your sales representative.

#### R.1.1 Writing the plugin

A plugin is a shared library that has specific functions and variables that will be called directly from Moab. The plugin conforms to a C language API. The API is specified through an include file: `moab-plugin.h`. This file must be included in the plugin code. The include file provides function definitions, structures and variables that will be used when communicating with Moab.

When you write the plugin, you need to ensure that the plugin code is robust. If the plugin crashes, Moab will crash. You will need to handle your own memory appropriately. If the plugin has memory leaks, Moab will have similar issues. If you want to maintain logs, the plugin will need to be responsible for its own logging.

##### *R.1.1.1 API and Data Structures*

The Application Programmer Interface (API) for the Moab Node Allocation Plugin consists of three data items and three entry points that must be supplied to Moab by the plugin.

Plugin Supplied Data	Description
<b>const char *PLUGIN_NAME = "Node Allocation plugin 1.1";</b>	This character pointer is used by Moab when logging information regarding the operation of the plugin.
<b>const char *PLUGIN_TYPE = PLUGIN_TYPE_NAME_NODEALLOCATION;</b>	This character pointer is used by Moab to verify the type of plugin. The value of this data is supplied by the <code>moab-plugin.h</code> source file. The plugin must set this as shown so that Moab does not attempt to use a plugin incorrectly. Moab uses this to determine whether the plugin API type is correct and to allow Moab to correctly communicate with the plugin.
<b>const char *PLUGIN_VERSION = PLUGIN_API_VERSION;</b>	This character pointer is used by Moab to verify the API version number. The value of this data is supplied by the <code>moab-plugin.h</code> source file. The plugin must set this as shown so that the correct version of the <code>moab-plugin.h</code> is supplied to Moab. Moab uses this to determine whether the API version is correct and to allow Moab to correctly communicate with the plugin.

Load Time API	Description
<b>initialize()</b>	<pre>int initialize(const char *name, void **data_handle)</pre> <p>The plugin must supply an <code>initialize()</code> entry point. This entry point is called for each use instance of the plugin. For example, if the plugin is used on two different partitions, the <code>initialize()</code> entry point will be called once for each partition.</p> <ul style="list-style-type: none"> <li>• Name — The name is the unique identifier which is used to distinguish multiple instances of the plugin and for logging. When configured globally, the name "ALL" will be given.</li> <li>• Data handle — The <code>data_handle</code> points to a location where the plugin should store a pointer to any internal data needed by the plugin between calls to the API. The actual format and structure of the data is up to the plugin. Moab will supply this pointer back to the plugin each time a plugin entry point is called. This data can provide context for the plugin usage instance.</li> </ul>
<b>Return codes</b>	<p>The <code>initialize()</code> entry point should return one of two return statuses as defined in <code>moab-plugin.h</code>:</p> <pre>#define PLUGIN_RC_SUCCESS 0 #define PLUGIN_RC_FAILURE 1</pre>

Load Time API	Description
<b>Gathering node info</b>	The initialize() entry point must gather any information about system nodes, their topology, interconnection, and configuration that it needs to make correct node allocations. Since Moab does not know what information the plugin may need, the plugin must gather this information itself.
<b>Memory considerations</b>	The plugin may allocate memory for temporary or persistent data as needed, but <i>must</i> de-allocate or return the memory when finished. Not returning memory can result in memory leaks and unstable operation on the part of Moab.
<b>Multiple access</b>	A given loaded plugin can be used by more than one partition. This means that the plugin must maintain its internal data in such a way that calls to the plugin for the separate partitions do not conflict. It is recommended that internal data be allocated and a pointer to the data be kept in the data_handle described above as opposed to using global or static variables. Any global or static data will be shared between possible multiple instances of the plugin.

Runtime API	Description
<b>node_allocate</b>	<pre>int node_allocate (     void          *data_handle,     const char    *job_name,     int           container_count,     nalloc_container_t container[])</pre> <p>The plugin must provide a node_allocate() entry point. This entry point is called each time Moab needs to determine where (on what nodes) a job will eventually run. Note that this entry point can be called many times before the job is actually scheduled to run.</p> <ul style="list-style-type: none"> <li>• Data structures — Moab uses C data structures to pass information and lists of nodes to the plugin and receive them back from the plugin. See moab-plugin.h for the definitions of these structures and for information on how they relate to one another.</li> </ul>

Runtime API	Description																	
<b>Operations</b>	A node allocation request consists of one or more requirements. Each of these requirements is provided within a “container” structure. The container has information regarding the requirement to be met, the count and list of all nodes that are available to meet the requirement and a place to return the list of nodes that the plugin has chosen to use for the job.																	
	<table border="1"> <thead> <tr> <th data-bbox="380 531 558 793">Command</th> <th data-bbox="558 531 656 793">Moab Job Task Count</th> <th data-bbox="656 531 753 793">Job Node Count</th> <th data-bbox="753 531 850 793">Job Tasks Per Node</th> <th data-bbox="850 531 948 793">Node CFG Procs</th> <th data-bbox="948 531 1045 793">Node AV-L Procs</th> <th data-bbox="1045 531 1143 793">Plugin Node Mapped TC</th> <th data-bbox="1143 531 1321 793">requirement -&gt;taskcount</th> <th data-bbox="1321 531 1430 793">return_node_count</th> </tr> </thead> </table>									Command	Moab Job Task Count	Job Node Count	Job Tasks Per Node	Node CFG Procs	Node AV-L Procs	Plugin Node Mapped TC	requirement ->taskcount	return_node_count
	Command	Moab Job Task Count	Job Node Count	Job Tasks Per Node	Node CFG Procs	Node AV-L Procs	Plugin Node Mapped TC	requirement ->taskcount	return_node_count									
	<b>Non-ExactNode</b>																	
	-l nodes=12	12	0	0	8	8	8	12	2									
	-l nodes=12:ppn=2	24	0	2	8	8	8	24	3									
	<b>ExactNode</b>																	
	-l nodes=4	4	4	0	8	8	1	4	4									
	-l nodes=4:ppn=2	8	4	2	8	8	2	8	4									
	-l nodes=12	12	0	0	8	6	6	12	2									
The duty of the plugin is to use the information that it has previously gathered (during the initialization) to select from the available nodes those that will best fulfill the requirements.																		
The basic algorithm is to consume all the taskcount and memory on each node until the consumed task count is greater than or equal to the container's task_count and memory requirements.																		
A job's taskcount is calculated differently based on the JOBNODEMATCHPOLICY parameter. By default, it isn't defined and <code>-l nodes=#</code> actually requests the number of tasks without respect to the number of nodes. In this case, the plugin should consume all the tasks of each chosen node until the taskcount is greater and/or equal to the container's taskcount requirement. The plugin is for node allocation and not task placement.																		

Runtime API	Description
	<p>When the JOBNODEMATCHPOLICY EXACTNODE is configured, then <code>-l nodes=#</code> means the job wants # of nodes with 1 task per node. In this case, the nodes passed to the plugin will have a taskcount that is mapped down to what the job can only use on that node. Each node's taskcount should be consumed on each node until the summed amount is equal to the container's requirement taskcount requirement.</p> <p>The following table shows how commands are interpreted by Moab and translated to the plugin and what is expected of the plugin.</p>
<b>Errors and return codes</b>	<p>The plugin may internally log any errors encountered and must return a success or error status as defined in <code>moab-plugin.h</code>:</p> <pre>#define PLUGIN_RC_SUCCESS 0 #define PLUGIN_RC_FAILURE 1</pre>
<b>Multiple access safe</b>	<p>The <code>node_allocate()</code> entry point must support multiple access as described above.</p>

Unload Time API	Description
<b>finish()</b>	<p><code>void finish(void *data_handle)</code>                      The plugin must supply a <code>finish()</code> entry point. This entry point is called when Moab is preparing to disable and/or unload an instance of the plugin.</p>
<b>Memory/resource cleanup</b>	<p>The plugin must de-allocate and free up any resources acquired either during the <code>initialize()</code> entry point or during any calls to the <code>node_allocate()</code> entry point. When the last entry point returns, there should be no allocated memory or other resources still in use by the plugin instance.</p>
<b>Multiple access safe</b>	<p>The <code>finish()</code> entry point must support multiple access as described above.</p>

### R.1.2 Moab configuration

The actual loading of a plugin is accomplished by specifying the plugin in the Moab configuration file, `moab.cfg`.

### R.1.2.1 Moab.cfg

We recommend that you store all Moab plugins in the `$MOABHOMEDIR/lib` directory (e.g., `/opt/moab/lib`) as shared libraries (`*.so`). The name of the actual plugin shared library file is up to the plugin developer, which means you must give the correct name in the `moab.cfg` file to form the absolute plugin filename.

If a plug-in's specified shared library filename starts with a forward slash (`/`), it is an absolute file path name and Moab simply uses it without alteration. For example, if a plugin's specified shared library filename is `/opt/moab/plugins/plugin.so`, Moab will use it as the absolute plugin file path name.

If a plugin's specified shared library filename does not start with a forward slash (`/`), it is a plugin name and Moab forms the plugin's absolute path name by concatenating the Moab home directory, `"/lib/lib"`, the specified plugin name, and `".so"` to obtain the absolute path name. For example, if the `$MOABHOMEDIR` environment variable contains `/opt/moab` and the plugin name is `plugin`, Moab will create `/opt/moab/lib/libplugin.so` and use it as the absolute plugin file path name.

### R.1.2.2 Syntax rules

In order for Moab to use a plugin for the Node Allocation policy, instead of a built-in Moab policy, you must configure the policy in the `moab.cfg` file with the value "PLUGIN:" followed by the plugin's shared library file name. The examples below assume the environment variable `$MOABHOMEDIR` has a value of `/opt/moab`. Note the use of relative and absolute plugin shared library file path names in the parameter value and how they affect Moab's construction of the full path name.

Partition	Plug-in Name	moab.cfg Parameter	Moab-derived Full Path Name
global	plugin.so	<b>NODEALLOCATIONPOLICY</b> <i>PLUGIN:plugin.so</i>	/opt/moab/lib/libplugin.so
global	/usr/local/plugins/plugin.so	<b>NODEALLOCATIONPOLICY</b> <i>PLUGIN:/usr/local/plugins/plugin.so</i>	/usr/local/plugins/plugin.so
abc	plugin.so	<b>PARCFG[abc]</b> <b>NODEALLOCATIONPOLICY</b> <i>=PLUGIN:plugin.so</i>	/opt/moab/lib/libplugin.so
xyz	/usr/local/plugins/plugin.so	<b>PARCFG[xyz]</b> <b>NODEALLOCATIONPOLICY=</b> <i>PLUGIN:/usr/local/plugins/plugin.so</i>	/usr/local/plugins/plugin.so

### R.1.2.3 Troubleshooting

There are several commands that can be used to confirm that the Plugin Node Allocation Policy was loaded properly.

**mschedctl -l**

`mschedctl -l` is used to print out Moab's in memory configurations. If the plugin policy, with its full path, doesn't show for the configured partition then Moab failed to load the partition. Note that when the **NODEALLOCATIONPOLICY** is configured globally, it is configured on the "ALL" partition.

```
$ mschedctl -l -v|grep ^NODEALLOCATIONPOLICY
NODEALLOCATIONPOLICY[ALL] PLUGIN:/opt/moab/lib/libfirstavailable.so
NODEALLOCATIONPOLICY[a]  PLUGIN:/opt/moab/lib/liblastavailable.so
NODEALLOCATIONPOLICY[b]  CONTIGUOUS
NODEALLOCATIONPOLICY[c]  PLUGIN:/opt/moab/lib/libfirstavailable.so
NODEALLOCATIONPOLICY[d]  [NONE]
```

### mdiag -C

`mdiag -C` is used to validate the `moab.cfg` configuration. With a plugin node allocation policy, Moab will validate that it can successfully load the plugin and that all of the required symbols are present.

```
$ mdiag -C
...
INFO: line #35 is valid: 'NODEALLOCATIONPOLICY PLUGIN:firstavailable'
INFO: line #36 is valid: 'PARCFG[a]NODEALLOCATIONPOLICY=PLUGIN:lastavailable'
INFO: line #37 is valid: 'PARCFG[b]NODEALLOCATIONPOLICY=CONTIGUOUS'
INFO: line #38 is valid: 'PARCFG[d]NODEALLOCATIONPOLICY=PLUGIN:firstavailable'
```

## Appendix S: Scalable Systems Software Specification

- [SSS Job Object Specification](#)
- [SSS Resource Management and Accounting Protocol Message Format](#)
- [SSS Node Object Specification](#)
- [SSS Resource Management and Accounting Protocol Wire Protocol](#)

### Scalable Systems Software Job Object Specification

SSS Job Object Specification  
Draft Release Version 3.1.0  
26 April 2011

Scott Jackson, PNNLStringDavid Jackson, Ames Lab  
Brett Bode, Ames Lab

### Status of This Memo

This document describes the job object to be used by Scalable Systems Software compliant components. It is envisioned for this specification to be used in conjunction with the SSSRMAP protocol with the job object passed in the Data field of Requests and Responses. Queries can be issued to a job-cognizant component in the form of modified XPATH expressions to the Get field to extract specific information from the job object as described in the SSSRMAP protocol.

## Abstract

This document describes the syntax and structure of the SSS job object. A job model is described that is flexible enough to support the specification of very simple jobs as well as jobs with elaborate and complex specification requirements in a way that avoids complex structures and syntax when it is not needed. The basic assumption is that a solitary job specification should be usable for all phases of the job lifecycle and can be used at submission, queuing, staging, reservations, quotations, execution, charging, accounting, etc. This job specification provides support for multi-step jobs, as well as jobs with disparate task descriptions. It accounts for operational requirements in a grid or meta-scheduled environment where the job is executed by multiple hosts in different administrative domains that support different resource management systems.

## Table of Contents

- [Scalable Systems Software Job Object Specification](#)
- [Table of Contents](#)
- [1.0 Introduction](#)
  - [1.1 Goals](#)
  - [1.2 Non-Goals](#)
  - [1.3 Examples](#)
    - [1.3.1 Very Simple Example](#)
    - [1.3.2 Moderate Example](#)
    - [1.3.3 Elaborate Example](#)
- [2.0 Conventions used in this document](#)
  - [2.1 Keywords](#)
  - [2.2 Table Column Interpretations](#)
  - [2.3 Element Syntax Cardinality](#)
- [3.0 The Job Model](#)
- [4.0 JobGroup Element](#)
  - [4.1 JobGroup Properties](#)
    - [4.1.1 Simple JobGroup Properties](#)
    - [4.1.2 Job](#)
    - [4.1.3 JobDefaults](#)
  - [4.2 JobGroup Reference](#)
- [5.0 Job and JobDefaults Element](#)

- [5.1 Job Properties](#)
  - [5.1.1 Simple Job Properties](#)
  - [5.1.2 Feature Element](#)
  - [5.1.3 OutputFile Element](#)
  - [5.1.4 ErrorFile Element](#)
  - [5.1.5 InputFile Element](#)
  - [5.1.6 NotificationList Element](#)
  - [5.1.7 ResourceLimit Element](#)
  - [5.1.8 Credentials](#)
  - [5.1.9 Environment Element](#)
    - [5.1.9.1 Variable Element](#)
  - [5.1.10 Node Element](#)
  - [5.1.11 TaskDistribution Element](#)
  - [5.1.12 Dependency Element](#)
  - [5.1.13 Consumable Resources](#)
  - [5.1.14 Resource Element](#)
  - [5.1.15 Extension Element](#)
  - [5.1.16 TaskGroup](#)
  - [5.1.17 TaskGroupDefaults](#)
- [5.2 Job Reference](#)
- [6.0 TaskGroup and TaskGroupDefaults Element](#)
  - [6.1 TaskGroup Properties](#)
    - [6.1.1 Simple TaskGroup Properties](#)
    - [6.1.2 Task](#)
    - [6.1.3 TaskDefaults](#)
  - [6.2 TaskGroup Reference](#)
- [7.0 Task and TaskDefaults Element](#)
  - [7.1 Task Properties](#)
    - [7.1.1 Simple Task Properties](#)
  - [7.2 Task Reference](#)
- [8.0 Property Categories](#)

- [8.1 Requested Element](#)
- [8.2 Delivered Element](#)
- [9.0 AwarenessPolicy Attribute](#)
- [10. References](#)
- [Appendix A](#)
- [Units of Measure Abbreviations](#)

## 1.0 Introduction

This specification proposes a standard XML representation for a job object for use by the various components in the SSS Resource Management System. This object will be used in multiple contexts and by multiple components. It is anticipated that this object will be passed via the Data Element of SSSRMAP Requests and Responses.

### 1.1 Goals

There are several goals motivating the design of this representation.

The representation needs to be inherently flexible. We recognize we will not be able to exhaustively include the ever-changing job properties and capabilities that constantly arise.

The representation should use the same job object at all stages of that job's lifecycle. This object will be used at job submission, queuing, scheduling, charging and accounting, hence it may need to distinguish between requested and delivered properties.

The design must account for the properties and structure required to function in a meta or grid environment. It needs to include the capability to support local mapping of properties, global namespaces, etc.

The equivalent of multi-step jobs must be supported. Each step (job) can have multiple logical task descriptions.

Many potential users of the specification will not be prepared to implement the complex portions or fine-granularity that others need. There needs to be a way to allow the more complicated structure to be added as needed while leaving more straightforward cases simple.

There needs to be guidance for how to understand a given job object when higher order features are not supported by an implementation, and which parts are required, recommended and optional for implementers to implement.

It needs to support composite resources.

It should include the ability to specify preferences or fuzzy requirements.

### 1.2 Non-Goals

Namespace considerations and naming conventions for most property values are outside of the scope of this document.

### 1.3 Examples

#### *Example 3-181: Very Simple Example*

This example shows a simple job object that captures the requirements of a simple job.

```

<Job>
  <Id>PBS.1234.0</Id>
  <State>Idle</State>
  <User>scottmo</User>
  <Executable>/bin/hostname</Executable>
  <Processors>16</Processors>
  <Duration>3600</Duration>
</Job>

```

### Example 3-182: Moderate Example

This example shows a moderately complex job object that uses features such as required versus delivered properties.

```

<Job>
  <Id>PBS.1234.0</Id>
  <Name>Heavy Water</Name>
  <Project>nwchemdev</Project>
  <User>peterk</User>
  <Application>NWChem</Application>
  <Executable>/usr/local/nwchem/bin/nwchem</Executable>
  <Arguments>-input basis.in</Arguments>
  <InitialWorkingDirectory>/home/peterk</InitialWorkingDirectory>
  <Machine>Colony</Machine>
  <QualityOfService>BottomFeeder</QualityOfService>
  <Queue>batch_normal</Queue>
  <State>Completed</State>
  <StartTime>1051557713</StartTime>
  <EndTime>1051558868</EndTime>
  <Charge>25410</Charge>
  <Requested>
    <Processors op="GE">12</Processors>
    <Memory op="GE" units="GB">2</Memory>
    <Duration>3600</Duration>
  </Requested>
  <Delivered>
    <Processors>16</Processors>
    <Memory metric="Average" units="GB">1.89</Memory>
    <Duration>1155</Duration>
  </Delivered>
  <Environment>
    <Variable name="PATH">/usr/bin:/home/peterk</Variable>
  </Environment>
</Job>

```

### Example 3-183: Elaborate Example

This example uses a job group to encapsulate a multi-step job. It shows this protocol's ability to characterize complex job processing capabilities. A component that processes this message is free to retain only that part of the information that it requires. Superfluous information can be ignored by the component or filtered out (by XSLT for example).

```

<JobGroup>
  <Id>workflow1</Id>
  <State>Active</State>
  <Name>ShuttleTakeoff</Name>
  <JobDefaults>
    <StagedTime>1051557859</StagedTime>
    <SubmitHost>asteroid.lbl.gov</SubmitHost>
    <SubmitTime>1051556734</SubmitTime>
    <Project>GrandChallenge18</Project>
    <GlobalUser>C=US,O=LBNL,CN=Keith Jackson</GlobalUser>
    <User>keith</User>
    <Environment>
      <Variable name="LD_LIBRARY_PATH"/>usr/lib</Variable>
      <Variable name="PATH"/>usr/bin:~/bin:</Variable>
    </Environment>
  </JobDefaults>
  <Job>
    <Id>fr15n05.1234.0</Id>
    <Name>Launch Vector Initialization</Name>
    <Executable>usr/local/gridphys/bin/lvcalc</Executable>
    <Queue>batch</Queue>
    <State>Completed</State>
    <Machine>SMP2.emsl.pnl.gov</Machine>
    <StartTime>1051557713</StartTime>
    <EndTime>1051558868</EndTime>
    <Quote>http://www.pnl.gov/SMP2#654321</Quote>
    <Charge units="USD">12.75</Charge>
    <Requested>
      <Duration>3600</Duration>
      <Processors>2</Processors>
      <Memory>1024</Memory>
    </Requested>
    <Delivered>
      <Duration>1155</Duration>
      <Processors consumptionRate="0.78">2</Processors>
      <Memory metric="Max">975</Memory>
    </Delivered>
    <TaskGroup>
      <TaskCount>2</TaskCount>
      <TaskDistribution type="TasksPerNode">1</TaskDistribution>
      <Task>
        <Node>node1</Node>
        <Process>99353</Process>
      </Task>
      <Task>
        <Node>node12</Node>
        <Process>80209</Process>
      </Task>
    </TaskGroup>
  </Job>
  <Job>
    <Id>fr15n05.1234.1</Id>
    <Name>3-Phase Ascension</Name>
    <Queue>batch_normal</Queue>
    <State>Idle</State>
    <Machine>Colony.emsl.pnl.gov</Machine>
    <Priority>1032847</Priority>
    <Hold>System</Hold>
    <StatusMessage>Insufficient funds to start job</StatusMessage>
    <Requested>

```

```

    <Duration>43200</Duration>
  </Requested>
</TaskGroup>
<TaskGroup>
  <TaskCount>1</TaskCount>
  <Name>Master</Name>
  <Executable>/usr/local/bin/stage-coordinator</Executable>
  <Memory>2048<Memory>
  <Resource name="License" type="ESSL2">1</Resource>
  <Feature>Jumbo-Frame</Feature>
</TaskGroup>
<TaskGroup>
  <Name>Slave</Name>
  <TaskDistribution type="Rule">RoundRobin</TaskDistribution>
  <Executable>/usr/local/bin/stage-slave</Executable>
  <NodeCount>4</NodeCount>
  <Requested>
    <Processors group="-1">12</Processors>
    <Processors conj="Or" group="1">16</Processors>
    <Memory>512</Memory>
    <Node aggregation="Pattern">fr15n.*</Node>
  </Requested>
</TaskGroup>
</Job>
</JobGroup>

```

## 2.0 Conventions Used in This Document

### 2.1 Keywords

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, RECOMMENDED, MAY, and OPTIONAL in this document are to be interpreted as described in [RFC2119](#).

### 2.2 Table Column Interpretations

The columns of the property tables in this document have the following meanings:

Element Name	Name of the XML element (xsd;element) see [DATATYPES]
<b>Type</b>	Data type defined by xsd (XML Schema Definition) as: <ul style="list-style-type: none"> <li>• String — xsd:string (a finite length sequence of printable characters)</li> <li>• Integer — xsd:integer (a signed finite length sequence of decimal digits)</li> <li>• Float — xsd:float (single-precision 32-bit floating point)</li> <li>• Boolean — xsd:boolean (consists of the literals “true” or “false”)</li> <li>• DateTime — xsd:int (a 32-bit unsigned long in GMT seconds since the EPOCH)</li> <li>• Duration — xsd:int (a 32-bit unsigned long measured in seconds)</li> </ul>
<b>Description</b>	Brief description of the meaning of the property

Element Name	Name of the XML element (xsd:element) see [DATATYPES]
<b>Appearance</b>	<p>An indication of whether the given property must appear in the parent element. It assumes the following meanings:</p> <ul style="list-style-type: none"> <li>• <b>MUST</b> — This property is <b>REQUIRED</b> when the parent is specified</li> <li>• <b>SHOULD</b> — This property is <b>RECOMMENDED</b> when the parent is specified.</li> <li>• <b>MAY</b> — This property is <b>OPTIONAL</b> when the parent is specified.</li> </ul>
<b>Compliance</b>	<p>An indication of the relative importance of supporting the given property.</p> <ul style="list-style-type: none"> <li>• <b>MUST</b> — A compliant implementation <b>MUST</b> support this property.</li> <li>• <b>SHOULD</b> — A compliant implementation <b>SHOULD</b> support this property.</li> <li>• <b>MAY</b> — A compliant implementation <b>MAY</b> support this property.</li> </ul>
<b>Categories</b>	<p>Some properties may be categorized into one of several categories. Letters in this column indicate that the given property can be classified in the following property categories.</p> <ul style="list-style-type: none"> <li>• <b>R</b> — This property can be encompassed in a Requested element.</li> <li>• <b>D</b> — This property can be encompassed in a Delivered element.</li> </ul>

### 2.3 Element Syntax Cardinality

Selected elements in the element syntax sections use regular expression wildcards with the following meanings:

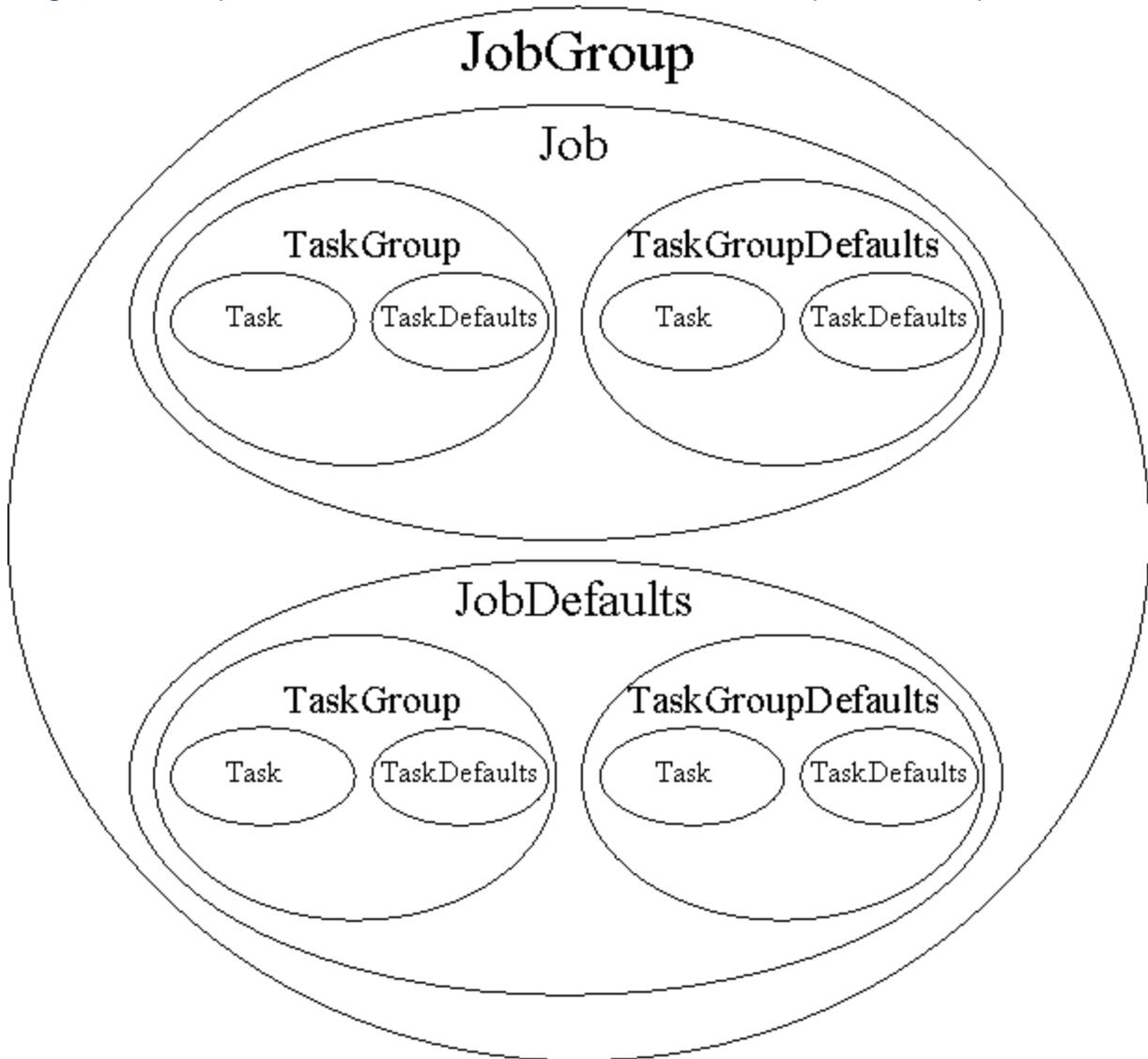
Wildcard	Description
*	Zero or more occurrences
+	One or more occurrences
?	Zero or one occurrences

The absence of one of these symbols implies exactly one occurrence.

## 3.0 The Job Model

The primary object within the job model is a job. A job can be thought of as a single schedulable entity and will be the object normally seen in job queues.

Image 3-16: JobGroup contains Job and JobDefaults, which contain TaskGroup and TaskGroupDefaults



Jobs with dependencies on other jobs may be submitted in a job group. Jobs within a job group form a DAG (directed acyclic graph) where the nodes are jobs and the edges represent dependencies on the status of previous jobs. A job group will consist of at least one job. A job group can optionally specify job defaults which are a set of job properties to be assumed by all jobs within the job group unless overridden within the job.

A job may consist of multiple tasks, which are the finest grained work unit and represent an endpoint for executing a given process instance. For example, a job that requests 3 nodes and 4 processors will have 4 tasks, two on one node and one on each of two nodes. Tasks may be grouped into task groups, which are logical aggregations of tasks and their common properties. Submit filters, prologs, epilogs, notification scripts, etc. run once only for each job. Whereas task groups function as logical descriptions of tasks and their properties, they also describe the number of such tasks and the nodes that they run on. As an example, a master task group (consisting of a single task) might ask for a node with a MATLAB license,

2GB of memory and an Internet connected network adapter while a slave task group (consisting of 12 tasks) could be targeted for nodes with more CPU bandwidth -- all within the same job and utilizing a common MPI ring. Tasks (and hence taskgroups) can have different executables or environments, specify different consumable resources or node properties. A job, therefore, may specify one or more task group. A job that does not specify an explicit task group is considered as having a single implicit task group. A job can optionally specify task group defaults which are a set of task group properties to be assumed by all task groups within the job unless overridden within a task group.

A task group may specify one or more tasks. A task group that does not specify an explicit task is considered as having a single implicit task. A task group can optionally specify task defaults which are a set of task properties to be assumed by all tasks within the task group unless overridden within a task.

## 4.0 JobGroup Element

A JobGroup is an optional element that aggregates one or more interdependent jobs. Some resource managers support the submission of job groups (multi-step jobs) and queries on the status of an entire job group.

- A compliant implementation MAY support this element.
- A JobGroup MUST specify one or more JobGroup Properties.
- A JobGroup MUST contain one or more Jobs.
- A JobGroup MAY contain zero or more JobsDefaults.

The following illustrates this element’s syntax:

```
C<JobGroup>
  <!-- JobGroup Properties -->+
  <Job/>+
  <JobDefaults/>?
</JobGroup>
```

### 4.1 JobGroup Properties

JobGroup Properties are properties that apply to the job group as a whole. These include the job group id, jobs and job defaults, and other simple optional job properties.

#### Simple JobGroup Properties

Simple (unstructured) job group properties are enumerated in the table below.

Table 3-7: Simple JobGroup Properties

Element Name	Type	Description	Appearance	Compliance
<b>CreationTime</b>	DateTime	Date and time that the job group was instantiated	MAY	MAY

Element Name	Type	Description	Appearance	Compliance
<b>Description</b>	String	Description of the job group	MAY	MAY
<b>Id</b>	String	Job group identifier	MUST	MUST
<b>Name</b>	String	Name of the job group	MAY	SHOULD
<b>State</b>	String	State of the job group as a whole. Valid states may include NotQueued, Unstarted, Active, and Completed.	MAY	SHOULD

*Job*

A job group **MUST** contain one or more jobs.

See the next section for element details.

*JobDefaults*

A job group **MAY** contain zero or one job defaults.

See the next section for element details.

**4.2 JobGroup Reference**

When a simple reference to a predefined job group is needed in an encapsulating element, a JobGroup element is used with the text content being the job group id:

```
<JobGroup> workflow1</JobGroup>
```

## 5.0 Job and JobDefaults Element

The Job and JobDefaults elements are of the same structure. A Job element encapsulates a job and may be expressed as a standalone object. A JobDefaults element may only appear within a JobGroup and represents the defaults to be taken by all jobs within the job group. Job properties in Job elements override any properties found in a sibling JobDefaults element.

- A compliant implementation **MUST** support the Job element.
- A compliant implementation **MAY** support the JobDefaults element only if it supports the JobGroup element.
- A job **MUST** specify one or more Job Properties.
- One or more TaskGroup elements **MAY** appear at this level.
- Zero or one TaskGroupDefaults elements **MAY** appear at this level.

The following illustrates this element’s syntax:

```
<Job>
  <!-- Job Properties -->+
  <TaskGroup/>*
  <TaskGroupDefaults/>?
</Job>
```

### 5.1 Job Properties

Job Properties apply to a particular job or as default properties to all jobs. They include the job id, job credentials, task groups, task group defaults, and other simple optional properties.

#### Simple Job Properties

Simple (unstructured) job properties are enumerated in the table below.

Table 3-8: Simple Job Properties

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Application</b>	String	Type of application such as Gaussian or Nwchem	MAY	MAY	

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Architecture</b>	String	Type architecture for the nodes on which this job must run	MAY	MAY	RD
<b>Arguments</b>	String	The arguments for the executable	MAY	SHOULD	
<b>Charge</b>	Float	The amount charged for the job	MAY	SHOULD	
<b>Checkpointable</b>	Boolean	Can this job be checkpointed?	MAY	MAY	
<b>CpuDuration</b>	Duration	Number of cpu seconds used by the job	MAY	SHOULD	
<b>DeadlineTime</b>	DateTime	Date and time that a job must end by	MAY	MAY	
<b>EligibleTime</b>	DateTime	Date and time that a job must start after	MAY	MAY	
<b>EndTime</b>	DateTime	Date and time that a job ended (independent of success or failure)	MAY	MUST	
<b>Executable</b>	String	Executable. This may be an absolute or relative path or a URI.*	MAY	MUST	

Element Name	Type	Description	Appearance	Compliance	Categories
<b>ExitCode</b>	Integer	Exit code for the job	MAY	SHOULD	
<b>GlobalJob</b>	String	Globally unique job identifier (possibly in the form of a URI)	MAY	SHOULD	
<b>Hold</b>	String	Hold(s) on the job. There may be multiple instances of this element if there is more than one ld on the job	MAY	SHOULD	
<b>InitialWorking-Directory</b>	String	Initial working directory	MAY	SHOULD	
<b>Interactive</b>	Boolean	Is this an interactive job?	MAY	SHOULD	
<b>Id</b>	String	A local job identifier assigned to the job by the local resource manager	MUST	MUST	
<b>Name</b>	String	Name of the job	MAY	SHOULD	
<b>State</b>	String	State of the job. Valid states may include Idle, Hold, Running, Suspended, or Completed	MAY	MUST	

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Type</b>	String	Type of job. Meaning of this extension property is context specific.	MAY	MAY	
<b>Machine</b>	String	Name of the system or cluster that runs the job	MAY	MUST	RD
<b>Network</b>	String	Type of network adapter required by the job	MAY	MAY	RD
<b>NodeCount</b>	Integer	Number of nodes used by the job	MAY	MUST	RD
<b>OperatingSystem</b>	String	Operating System required by the job	MAY	MAY	RD
<b>Partition</b>	String	Name of the partition in which the job should run	MAY	MAY	RD
<b>Priority</b>	Integer	Current queue priority (or rank)for the job	MAY	SHOULD	
<b>QualityOfService</b>	String	Name of the Quality of Service (QoS)	MAY	SHOULD	RD
<b>Queue</b>	String	Name of the Queue (or class) that the job runs in	MAY	SHOULD	RD

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Quote</b>	String	Identifier for a guaranteed charge rate quote obtained by the job	MAY	MAY	
<b>Reservation</b>	String	Identifier for a reservation used by the job	MAY	MAY	RD
<b>ReservationTime</b>	DateTime	Date and time that a reservation was placed for the job	MAY	MAY	
<b>ResourceManagerType</b>	String	Type of resource manager required to run this job	MAY	MAY	RD
<b>Restartable</b>	Boolean	Can this job be restarted?	MAY	MAY	
<b>Shell</b>	String	Specified the shell necessary to interpret the job script	MAY	MAY	
<b>StagedTime</b>	DateTime	Date and time that a job was staged to the local resource management system	MAY	MAY	
<b>StartCount</b>	Integer	Number of times the scheduler tried to start the job	MAY	MAY	

Element Name	Type	Description	Appearance	Compliance	Categories
<b>StartTime</b>	DateTime	Date and time that the job started	MAY	MUST	
<b>StatusMessage</b>	String	Natural language message that can be used to provide detail on why a job failed, isn't running, etc.	MAY	SHOULD	
<b>SubmitTime</b>	DateTime	Date and time that a job was submitted	MAY	SHOULD	
<b>SubmitHost</b>	String	FQDN of host where the job was submitted from	MAY	SHOULD	
<b>Suspendable</b>	Boolean	Can this job be suspended?	MAY	MAY	
<b>SuspendDuration</b>	Integer	Number of seconds the job was in the Suspended state	MAY	MAY	
<b>TimeCategory</b>	String	This allows the specification of shifts like PrimeTime for charging purposes	MAY	MAY	
<b>Duration</b>	Duration	Number of seconds in the Running state	SHOULD	MUST	RD

\* The Executable may be a script or a binary executable. If it is already on the target system it may be referenced by an absolute or relative pathname (relative to InitialWorkingDirectory). If it is passed with the job in a File object (see SSSRMAP), it can be referenced by an absolute or relative URI. An absolute

URI would specify a URL where the file can be downloaded (like with `wget`). A relative URI is specified by preceding an identifier by a pound sign, as in

```
<Executable>#Script</Executable>
```

It will be found in a `File` object included along with the `Job` object with the `Script` as an identifier, as in

```
<File id="Script">echo hello world</File>
```

### Feature Element

The `Feature` element connotes an arbitrary named feature of a node.

- A compliant implementation **SHOULD** support this element.
- This element **MAY** appear zero or one times within a given set of `Job Properties`.
- This element is of type `String`.
- This element **MAY** have an `aggregation` attribute of type `String` that provides a way to indicate multiple values with a single expression. A compliant implementation **MAY** support the `aggregation` attribute if the `Feature` element is supported. Possible values for this attribute include:
  - `List` — a comma-separated list of features
  - `Pattern` — a regular expression (perl5) matching desired features
- If an `aggregation` attribute is specified with the value of `List`, this element **MAY** also have a `delimiter` attribute of type `String` that indicates what delimiter is used to separate list elements. The default list delimiter is a comma.
- This element **MAY** be categorized as a requested or delivered property by being encompassed by the appropriate element.

The following is an example of a feature element:

```
<Feature aggregation="List">feature1,feature2</Feature>
```

### OutputFile Element

The `OutputFile` element specifies the name of the file to which the output stream (`stdout`) from the job will be written.

- This element's character content is the name of the file. If this element is omitted or it is empty, then an appropriate output file is auto-determined by the queuing system.
- This element **MAY** have a `redirectList` attribute which is a comma-separated list of output redirection attributes of type `String`. A compliant implementation **SHOULD** support this attribute if `OutputFile` is supported. Possible values for this attribute include:
  - `Append` — opens the output file for append
  - `Close` — closes and discards the output stream
  - `Flush` — output is written to output file as it is generated

- Keep — leave the output file on the execution host
- Merge — merges the output stream into the error stream

Note that when using the `redirectList` attributes, the cumulative effect of the `ErrorFile` and `OutputFile` directives may be order dependent.

The following is an example of an `OutputFile` element:

```
<OutputFile redirectList="Append">~/myjob.out</OutputFile>
```

### *ErrorFile Element*

The `ErrorFile` element specifies the name of the file to which the error stream (`stderr`) from the job will be written.

- This element's character content is the name of the file. If this element is omitted or it is empty, then an appropriate error file is auto-determined by the queuing system.
- This element MAY have a `redirectList` attribute which is a comma-separated list of error redirection attributes of type `String`. A compliant implementation SHOULD support this attribute if `ErrorFile` is supported. Possible values for this attribute include:
  - Close — closes and discards the error stream
  - Append — opens the error file for append
  - Flush — output is written to output file as it is generated
  - Keep — leave the output file on the execution host
  - Merge — merges the error stream into the output stream

Note that when using the `redirectList` attributes, the cumulative effect of the `ErrorFile` and `OutputFile` directives may be order dependent.

The following is an example of an `ErrorFile` element:

```
<ErrorFile redirectList="Merge"></ErrorFile>
```

### *InputFile Element*

The `InputFile` element specifies the name of the file from which the input stream (stdin) for the job will be read.

- This element's character content is the name of the file. If this element is omitted or it is empty, then an appropriate input file is auto-determined by the queuing system.
- This element MAY have a `redirectList` attribute which is a comma-separated list of input attributes of type String. A compliant implementation SHOULD support this attribute if `InputFile` is supported. Possible values for this attribute include:
  - `Close` — closes and discards the input stream

The following is an example of an `InputFile` element:

```
<InputFile redirectList="Close"></InputFile>
```

### *NotificationList Element*

The `NotificationList` element specifies the job-related events or conditions for which a notification will be sent.

- This element's character content is a comma-separated list of events or conditions for which a notification should be sent. Possible values for the elements of this list include:
  - `JobStart` — send a notification when the job starts
  - `JobEnd` — send a notification when the job ends
  - `All` — send notifications for all notifiable events
  - `None` — do not send notifications for any events
- This element MAY have a `uri` attribute of type String which indicates where the notification is to be sent. A compliant implementation MAY support this attribute if `NotificationList` is supported. The `uri` is in the format: `[scheme://]authority` with the scheme being `smtp` and the authority being an email address by default.

The following is an example of a `NotificationList` element:

```
<NotificationList uri="smith@business.com">JobStart,JobEnd</NotificationList>
```

### *ResourceLimitElement*

The `ResourceLimit` element represents a resource limit with its name and value.

- This element MUST have a `name` attribute of type String. A compliant implementation MUST support the `name` attribute if `ResourceLimit` is supported.
- This element MAY have a `type` attribute of type String that may have the values `Hard` or `Soft`. If the limit is enforced by the operating system, a hard limit is one that cannot be increased once

it is set while a soft limit may be increased up to the value of the hard limit. If the type attribute is omitted, both the soft and hard limits are set.

- This element's character content is the resource limit's value.

Some typical names include:

Name	Description
<b>CoreFileSize</b>	Maximum core file size
<b>CpuTime</b>	CPU time in seconds
<b>DataSegSize</b>	Maximum data size
<b>FileSize</b>	Maximum file size
<b>MaxMemorySize</b>	Maximum resident set size
<b>MaxProcesses</b>	Maximum number of processes
<b>MaxSwap</b>	Virtual memory limit
<b>MaxMemLock</b>	Maximum locked-in-memory address space
<b>MaxProcessors</b>	Maximum processors
<b>MaxMemory</b>	Maximum memory
<b>MaxDisk</b>	Maximum disk space
<b>MaxNetwork</b>	Maximum network bandwidth
<b>MaxFileIO</b>	Maximum file i/o
<b>OpenFiles</b>	Maximum number of open files
<b>Stacksize</b>	Maximum stack size

The following is an example of a ResourceLimit element:

```
<ResourceLimit name="CpuTime">1000000</ResourceLimit>
```

### Credentials

Credentials are a special group of job properties that characterize an authenticated token or id. They can be categorized in both requested and delivered forms.

Credential job properties are enumerated in the table below.

Table 3-9: Credential Job Properties

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Project</b>	String	Name of the Project or Charge Account	MAY	SHOULD	RD
<b>GlobalUser</b>	String	Globally unique user identifier. This may be an X.509 DN for example	MAY	SHOULD	RD
<b>Group</b>	String	Name of the local group id	MAY	MAY	RD
<b>User</b>	String	Name of the local user id for the job	MAY	MUST	RD

### Environment Element

The Environment element encapsulates environment variables.

- This element MAY have an export attribute of type Boolean that which if set to `True` indicates that all environment variables in the context of the job submission process should be exported in the job’s execution environment.
- A compliant implementation SHOULD support this element.
- An Environment element MAY appear zero or one times within a given set of Job (or TaskGroup) Properties.
- An Environment element MAY contain one or more Variable elements.

The following illustrates this element’s syntax:

```
<Environment>
  <Variable/>+
</Environment>
```

### Variable Element

The Variable element represents an environment variable with its name and value.

This element **MUST** have a `name` attribute of type `String`. A compliant implementation **MUST** support the `name` attribute if `Variable` is supported. This element's character content is the environment variable's value.

The following is an example of a `Variable` element:

```
<Variable name="PATH"/usr/bin:/home/sssdemo</Variable>
```

### Node Element

The `Node` element represents a node.

- A compliant implementation **SHOULD** support this element.
- This element **MAY** appear zero or one times within a given set of Job Properties.
- This element is of type `String`.
- This element **MAY** have an `aggregation` attribute of type `String` that provides a way to indicate multiple values with a single expression. A compliant implementation **MAY** support the `aggregation` attribute if the `Feature` element is supported. Possible values for this attribute include:
  - `List` - a comma-separated list of features
  - `Pattern` - a regular expression (perl5) matching desired features
  - `Range` - a range of nodes of the form: `<prefix>[5-23,77]`
- If an `aggregation` attribute is specified with the value of `List`, this element **MAY** also have a `delimiter` attribute of type `String` that indicates what delimiter is used to separate list elements. The default list delimiter is a comma.
- This element **MAY** have a `count` attribute of type `Integer` that indicates the instance count of the specified node(s).
- This element **MAY** be categorized as a requested or delivered property by being encompassed by the appropriate element.

The following is an example of a `Node` element:

```
<Node aggregation="Pattern">node[1-5]</Node>
```

### TaskDistribution Element

The `TaskDistribution` element describes how tasks are to be mapped to nodes. This mapping may be expressed as a rule name, a task per node ratio or an arbitrary geometry.

- A compliant implementation SHOULD support this element.
- This element MAY appear zero or one times in a given set of Job (or TaskGroup) Properties.
- This element is of type String.
- This element MAY have a `type` attribute of type String that provides a hint as to the type of mapping guidance provided. It may have values including `Rule`, `TasksPerNode`, `ProcessorsPerTask` or `Geometry`. A compliant implementation MAY support the `type` attribute if the `TaskDistribution` element is supported.
- It is possible to use `Processors`, `NodeCount` and `TaskCount` elements to specify a set of mutually contradictory task parameters. When this occurs, components are responsible for resolving conflicting requirements.

The following are three examples of a `TaskDistribution` element:

```
<TaskDistribution type="TasksPerNode">2</TaskDistribution>
<TaskDistribution type="Rule">RoundRobin</TaskDistribution>
<TaskDistribution type="Geometry">{1,4}{2}{3,5}</TaskDistribution>
```

### Dependency Element

The `Dependency` element allows a job's execution to depend on the status of other jobs. In a job group (multi-step job), some jobs may delay execution until the failure or success of other jobs creating in general a Directed Acyclic Graph relationship between the jobs. This element's content is of type String and represents the job that the current job is dependent upon. Since a job may have two or more dependencies, this element may appear more than once in a given job scope. A compliant implementation SHOULD support this element if job groups are supported.

- A compliant implementation SHOULD support this element.
- This element MAY appear zero or more times in a given set of Job (or TaskGroup) Properties.
- This element is of type String and contains the JobId that the current job is dependent upon.
- This element MAY have a `condition` attribute of type String that indicates the basis for determining when the current job executes in relation to the specified job. A compliant implementation MUST support this attribute if this element is supported. Possible values for this attribute include:
  - `OnSuccess` this job should run after the referenced job only if it completes successfully (this is the default if the `type` attribute is omitted)

- OnFailure this job should run after the referenced job only if it fails
- OnExit this job should run after the referenced job exits
- If the `condition` attribute is equal to OnExit, this element MAY have a `code` attribute of type Integer that indicates the exit code that will trigger this job to run. If the `code` attribute is omitted, then the current job should run after the referenced job for any exit status.
- This element MAY have a `designator` attribute of type String that indicates that indicates the property of the job that identifies it as the dependent job. A compliant implementation MAY support this attribute if this element is supported. Possible values for this attribute include:
  - JobId the job this job is dependent upon is specified by JobId (this is the default if the `designator` attribute is omitted)
  - JobName the job(s) this job is dependent upon are specified by JobName

The following is an example of a Dependency element:

```
<Dependency condition="OnSuccess" designator="JobId">PBS.1234.0</Dependency>
```

### Consumable Resources

Consumable Resources are a special group of properties that can have additional attributes and can be used in multiple contexts. In general a consumable resource is a resource that can be consumed in a measurable quantity.

- A consumable resource MAY have a `context` attribute of type String that indicates the sense in which the resource is used. A compliant implementation MAY support this attribute. Possible values for this attribute include:
  - Configured — run this task only on nodes having the specified configured resources
  - Available — run this task only on nodes having the specified available resources. (this is the default if the `context` attribute is omitted)
  - Used — the task used the indicated resources (this is analogous to being including in a Delivered block)
  - Dedicated — the indicated amount of the resource should be dedicated to the task
- A consumable resource MAY have a `units` attribute that is of type String that specifies the units by which it is being measured. If this attribute is omitted, a default unit is implied. A compliant implementation MAY support this attribute if the element is supported.
- A consumable resource MAY have a `metric` attribute that is of type String that specifies the type of measurement being described. For example, the measurement may be a Total, an Average, a Min or a Max. A compliant implementation MAY support this attribute if the element is supported.
- A consumable resource MAY have a `duration` attribute of type Duration that indicates the amount of time for which that resource was used. This need only be specified if the resource was

used for a different amount of time than the duration for the job. A compliant implementation MAY support this attribute if the element is supported.

- A consumable resource MAY have a `consumptionRate` attribute of type Float that indicates the average percentage that a resource was used over its duration. For example, an overbooked SMP running 100 jobs across 32 processors may wish to scale the usage and charge by the average fraction of processor usage actually delivered. A compliant implementation MAY support this attribute if the element is supported.
- A consumable resource MAY have a `dynamic` attribute of type Boolean that indicates whether the resource allocated for this job should be allowed to grow or shrink dynamically. For example, if processors is specified with `dynamic` equal to True, the job may be dynamically allocated more processors as they become available. The growth bounds can be indicated via the `op` attribute which is inherited when a consumable resource element is encapsulated within a *Requested* element. A compliant implementation MAY support this attribute if the element is supported.

A list of simple consumable resources is listed in the table below.

Table 3-10: Simple Consumable Resources

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Disk</b>	Float	Amount of disk	MAY	SHOULD	RD
<b>Memory</b>	Float	Amount of memory	MAY	SHOULD	RD
<b>Network</b>	Float	Amount of network	MAY	MAY	RD
<b>Processors</b>	Integer	Number of processors	MAY	MUST	RD
<b>Swap</b>	Float	Amount of virtual memory	MAY	MAY	RD

The following are two examples for specifying a consumable resource:

```
<Memory metric="Max" units="GB">483</Memory>
<Processors duration="1234" consumptionRate="0.63">4</Processors>
```

### Resource Element

In addition to the consumable resources enumerated in the above table, an extensible consumable resource is defined by the Resource element.

- A compliant implementation SHOULD support this element.
- This element MAY appear zero or more times within a given set of job (or task group) properties.
- Like the other consumable resources, this property MAY be categorized as a requested or delivered property by being encompassed in the appropriate element.
- This element is of type Float.
- This element shares the *same properties and attributes as the other consumable resources* but it requires an additional name (and optional type) attribute to describe it.
- It MUST have a name attribute of type String that indicates the type of consumable resource being measured. A compliant implementation MUST support this attribute if the element is supported.
- It MAY have a type attribute of type String that distinguishes it within a general resource class. A compliant implementation SHOULD support this attribute if the element is supported.

The following are two examples for specifying a Resource element:

```
<Resource name="License" type="MATLAB">1</Resource>
<Resource name="Telescope" type="Zoom2000" duration="750" metric="KX">10</Resource>
```

### Extension Element

The Extension element provides a means to pass extensible properties with the job object.

Some applications may find it easier to use a named extension property than discover and handle elements they do not understand or anticipate by name.

- A compliant implementation MAY support this element.
- This element MUST have a name attribute of type String that gives the extension property's name. A compliant implementation MUST support this attribute if this element is supported.
- This element MAY have a type attribute of type String that characterizes the context within which the property should be understood. A compliant implementation SHOULD support this attribute if this element is supported.
- This element's character content, which is of type String, is the extension property's value.

The following is an example of an Extension element:

```
<Extension type="Scheduler" name="Restartable">true</Extension>
```

### *TaskGroup*

A job MAY specify one or more task groups.

See the next section for element details.

### *TaskGroupDefaults*

A job MAY specify zero or more task group defaults.

See the next section for element details.

## 5.2 Job Reference

When a simple reference to a predefined job is needed in an encapsulating element, a Job element is used with the text content being the job id:

```
<Job> job123</Job>
```

## 6.0 TaskGroup and TaskGroupDefaults Element

The `TaskGroup` and `TaskGroupDefaults` elements have the same structure. A `TaskGroup` element aggregates tasks. A `TaskGroupDefaults` element may only appear within a Job (or JobDefaults) and represents the defaults to be taken by all task groups within the job. Task group properties in `TaskGroup` elements override any properties found in a sibling `TaskGroupDefaults` element.

- A compliant implementation MAY support the `TaskGroup` element.
- A compliant implementation MAY support the `TaskGroupDefaults` element.
- A task group MUST specify one or more `TaskGroup` Properties.
- One or more `Task` elements MAY appear at this level.
- Zero or one `TaskDefaults` elements MAY appear at this level.

The following illustrates this element's syntax:

```
<TaskGroup>
  <!-- TaskGroup Properties -->+
  <!-- Job Properties -->*
  <Task>+
  <TaskDefaults>?
</TaskGroup>
```

### 6.1 TaskGroup Properties

`TaskGroup` Properties apply to a particular task group or as default properties to encompassed task groups. These properties include the task group id, its tasks, task defaults, and other simple task group properties.

### Simple TaskGroup Properties

Simple (unstructured) task group properties are enumerated in Table 6.

Table 3-11: Simple TaskGroup Properties

Element Name	Type	Description	Appearance	Compliance	Categories
<b>TaskCount</b>	Integer	Number of tasks in this taskgroup	MAY	MUST	
<b>Id</b>	String	A task group identifier unique within the job	MAY	MAY	
<b>Name</b>	String	A task group name (such as Master)	MAY	SHOULD	

#### Task

A task group MAY specify zero or more tasks.

See the next section for element details.

#### TaskDefaults

A task group MAY specify zero or more task defaults.

See the next section for element details.

## 6.2 TaskGroup Reference

When a simple reference to a predefined task group is needed in an encapsulating element, a TaskGroup element is used with the text content being the task group id:

```
<TaskGroup> tgl</TaskGroup>
```

## 7.0 Task and TaskDefaults Element

The `Task` and `TaskDefaults` elements have the same structure. A `Task` element contains information specific to a task (like the process id or the host it ran on). A `TaskDefaults` element may only appear within a `TaskGroup` (or `TaskGroupDefaults`) element and represents the defaults for all tasks within the task group. Task properties in `Task` elements override any properties found in a sibling `TaskDefaults` element.

- A compliant implementation MAY support the `TaskGroup` element.
- A compliant implementation MAY support the `TaskGroupDefaults` element.
- A task group MUST specify one or more `TaskGroup` Properties.
- One or more `Task` elements MAY appear at this level.
- Zero or one `TaskDefaults` elements MAY appear at this level.

The following illustrates this element’s syntax:

```
<Task>
  <!-- Task Properties -->+
  <!-- Job Properties -->*
</Task>
```

### 7.1 Task Properties

Task Properties are properties that apply to a particular task or as default properties to encompassed tasks. These properties include the task id and other task properties.

#### Simple Task Properties

Simple (unstructured) task properties are enumerated in the table below.

Table 3-12: Simple Task Properties

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Node</b>	String	Name of the node this task ran on	MAY	MUST	
<b>Session</b>	Integer	Session id for the task group or job	MAY	MAY	

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Id</b>	String	A task identifier unique within the taskgroup	MAY	MAY	

## 7.2 Task Reference

When a simple reference to a predefined task is needed in an encapsulating element, a `Task` element is used with the text content being the task id:

```
<Task>1</Task>
```

## 8.0 Property Categories

Certain properties need to be classified as being in a particular category. This is done when it is necessary to distinguish between a property that is requested and a property that was delivered. When no such distinction is necessary, it is recommended that the property not be enveloped in one of these elements. In general, a property should be enveloped in a category element only if it is expected that the property will need to be attributed to more than one property category, or if it needs to make use of some of the special attributes inherited from the category.

### 8.1 Requested Element

A requested property reflects properties as they were requested. A disparity might occur between the requested value and the value delivered if a preference was expressed, if multiple options were specified, or if ranges or pattern matching was specified.

- A compliant implementation SHOULD support this element.

The following illustrates the syntax of this element:

```
<Requested>
  <!-- Requested Properties -->+
</Requested>
```

The following describes the attributes and elements for the example above:

```
/Requested
```

This element is used to encapsulate requested properties.

```
/Requested/<Requested Property>
```

Requested properties appear at this level.

Requested Properties inherit some additional attributes.

- A requested property MAY have an `op` attribute of type String that indicates a conditional operation on the value. A compliant implementation SHOULD support this attribute. Valid values for the `op` attribute include EQ meaning equals (which is the default), NE meaning not equal, LT meaning less than, GT meaning greater than, LE meaning less than or equal to, GE meaning greater than or equal to, Match which implies the value is a pattern to be matched.
- A requested property MAY have a `conj` attribute of type String that indicates a conjunctive relationship with the previous element. A compliant implementation MAY support this attribute. Valid values for the `conj` attribute include And (which is the default), Or, Nand meaning and not, and Nor meaning or not.
- A requested property MAY have a `group` attribute of type Integer that indicates expression grouping and operator precedence much like parenthetical groupings. A compliant implementation MAY support this attribute. A positive grouping indicates the number of nested expressions being opened with the property while a negative grouping indicates the number of nested expressions being closed with the property.
- A requested property MAY have a `preference` attribute of type Integer that indicates a preference for the property along with a weight (the weights are taken as a ratio to the sum of all weights in the same group). A compliant implementation MAY support this attribute. If a group of positive valued preference alternatives are specified, at least one of the preferences must be satisfied for the job to run. If a group of negative valued preferences are specified, the preferences will try to be met according to their weights but the job will still run even if it can't satisfy any of the preferred properties. (Weight ranking can be removed by making all weights the same value (1 or -1 for example).
- A requested property MAY have a `performanceFactor` attribute of type Float that provides a hint to the scheduler of what performance tradeoffs to make in terms of resources and start time. A compliant implementation MAY support this attribute.

The following are four examples of using Requested Properties:

```

<Requested>
  <Processors op="GE">8</Processors>
  <Processors op="LE">16</Processors>
  <Duration>3600</Duration>
</Requested>
<Requested>
  <NodeCount>1</NodeCount>
  <Node aggregation="Pattern">fr15.*</Node>
</Requested>
<Requested>
  <User group="1">scottmo</User>
  <Account group="-1">mscfops</Account>
  <User conj="Or" group="1">amy</User>
  <Account group="-1">chemistry</Account>
</Requested>
<Requested>
  <Memory preference="2">1024</Memory>
  <Memory preference="1">512</Memory>
</Requested>

```

## 8.2 Delivered Element

A delivered property reflects properties as they were actually utilized, realized or consumed. It reflects the actual amounts or values that are used, as opposed to a limit, choice or pattern as may be the case with a requested property.

- A compliant implementation SHOULD support this element.

The following illustrates the syntax of this element:

```
<Delivered>
  <!-- Delivered Properties -->+
</Delivered>
```

The following describes the attributes and elements for the example above:

```
/Delivered
```

This element is used to encapsulate delivered properties.

```
/Delivered/<Delivered Property>
```

Delivered properties appear at this level.

Delivered Properties inherit some additional attributes.

- A delivered property MAY have a `group` attribute of type Integer that indicates expression grouping and operator precedence much like parenthetical groupings. A compliant implementation MAY support this attribute. A positive grouping indicates the number of nested expressions being opened with the property while a negative grouping indicates the number of nested expressions being closed with the property. The purpose of this attribute would be to logically group delivered properties if they were used in certain aggregations (like a job that spanned machines).

The following are the same four examples distinguishing the delivered amounts and values:

```

<Delivered>
  <Processors>12</Processors>
  <Duration>1234</Duration>
</Delivered>
<Delivered>
  <Node>fr15n03</Node>
</Delivered>
<Delivered>
  <User>scottmo</User>
  <Account>mscfops</Account>
</Delivered>
<Delivered>
  <Memory>1024</Memory>
</Delivered>

```

## 9.0 AwarenessPolicy Attribute

A word or two should be said about compatibility mechanisms. With all the leeway in the specification with regard to implementing various portions of the specification, problems might arise if an implementation simply ignores a portion of a job specification that is critical to the job function in certain contexts. Given this situation, it might be desirable in some circumstances for jobs to be rejected by sites that fail to fully support that job's element or attributes. At other times, it might be desirable for a job to run, using a best-effort approach to supporting unimplemented features. Consequently, we define an `awarenessPolicy` attribute which can be added as an optional attribute to the Job element or any other containment or property element to indicate how the property (or the default action for the elements that the containment element encloses) must react when the implementation does not understand an element or attribute.

An awareness policy of `Reject` will cause the server to return a failure if it receives a client request in which it does not support an associated element name or attribute name or value. It is reasonable for an implementation to ignore (not even look for) an element or attribute that would not be critical to its function as long as ignoring this attribute or element would not cause an incorrect result. However, any element or attribute that was present that would be expected to be handled in a manner that the implementation does not support must result in a failure.

An awareness policy of `Warn` will accept the misunderstood element or attribute and continue to process the job object on a best effort basis. However a warning **MUST** be sent (if possible) to the requestor enumerating the elements and attributes that are not understood.

An awareness policy of `Ignore` will accept the unsupported element or attribute and continue to process the job object on a best effort basis. The action could be to simply ignore the attribute.

- This name of this attribute is `awarenessPolicy`.
- This attribute is of type `String`.
- This attribute can have values of `Reject`, `Warn` or `Ignore`.
- A compliant implementation **MAY** support this attribute.

- An implementation that does not support an attribute **MUST** reject any job object which contains elements or attributes that it does not support. Furthermore, it **SHOULD** return a message to the requestor with an indication of the element or attribute name it did not understand.
- This attribute **MAY** be present in a property or containment element.
- If an implementation does support the attribute, but it is absent, the default value of `Reject` is implied.
- Individual elements in the job object may override the containing object's awareness policy default by including this attribute. For example, a job might specify an `awarenessPolicy` of `Reject` at its root (the `Job` element) but may want to allow a particular subset of elements or attributes to be ignored if not understood. Conversely, a job with a default `awarenessPolicy` of `Ignore` might want to classify a subset of its optional elements as `Reject` if they are indispensable to its correct interpretation. An implementation can opt to check or not check for this attribute at any level it wants but must assume a `Reject` policy for any elements it does not check.

## 10.0 References

### ISO 8601

ISO (International Organization for Standardization). Representations of dates and times, 1988-06-15. <http://www.iso.ch/markete/8601.pdf>

### DATATYPES

XML Schema Part 2: Datatypes. Recommendation, 02 MAY 2001. <http://www.w3.org/TR/xmlschema-2/>

## Appendix A

### Units of Measure Abbreviations

Abbreviation	Definition	Quantity
<b>B</b>	byte	1 byte
<b>KB</b>	Kilobyte	2 <sup>10</sup> bytes
<b>MB</b>	Megabyte	2 <sup>20</sup> bytes

Abbreviation	Definition	Quantity
<b>GB</b>	Gigabyte	2 <sup>30</sup> bytes
<b>TB</b>	Terabyte	2 <sup>40</sup> bytes
<b>PB</b>	Petabyte	2 <sup>50</sup> bytes
<b>EB</b>	Exabyte	2 <sup>60</sup> bytes
<b>ZB</b>	Zettabyte	2 <sup>70</sup> bytes
<b>YB</b>	Yottabyte	2 <sup>80</sup> bytes
<b>NB</b>	Nonabyte	2 <sup>90</sup> bytes
<b>DB</b>	Doggabyte	2 <sup>100</sup> bytes

## Scalable Systems Software Resource Management and Accounting Protocol (SSSRMAP) Message Format

Resource Management Interface Specs

Release v. 3.0.4

18 JUL 2005

Scott Jackson  
Brett Bode  
David Jackson  
Kevin Walker

### Status of This Memo

This is a specification defining an XML message format used between Scalable Systems Software components. It is intended that this specification will continue to evolve as these interfaces are implemented and thoroughly tested by time and experience.

### Abstract

This document is a specification describing a message format for the interaction of resource management and accounting software components developed as part of the Scalable Systems Software Center. The SSSRMAP Message Format defines a request-response syntax supporting both functional and object-oriented messages. The protocol is specified in XML Schema Definition. The message elements defined in this specification are intended to be framed within the Envelope and Body elements defined in the SSSRMAP Wire Protocol specification document.

## Table of Contents

- [1.0 Introduction](#)
- [2.0 Conventions Used in this Document](#)
  - [2.1 Keywords](#)
  - [2.2 XML Case Conventions](#)
  - [2.3 Schema Definitions](#)
- [3.0 Encoding](#)
  - [3.1 Schema Header and Namespaces](#)
  - [3.2 Element Descriptions](#)
    - [3.2.1 The Request Element](#)
    - [3.2.2 The Object Element](#)
    - [3.2.3 The Get Element](#)
    - [3.2.4 The Set Element](#)
    - [3.2.5 The Where Element](#)
    - [3.2.6 The Option Element](#)
    - [3.2.7 The Data Element](#)
    - [3.2.8 The File Element](#)
    - [3.2.9 The Count Element](#)
    - [3.2.10 The Response Element](#)
    - [3.2.11 The Status Element](#)
    - [3.2.12 The Value Element](#)
    - [3.2.13 The Code Element](#)
    - [3.2.14 The Message Element](#)
  - [3.3 Modified XPATH Expressions](#)
    - [3.3.1 Sample Modified XPATH expressions](#)
  - [3.4 Examples](#)
    - [3.4.1 Sample Requests](#)
    - [3.4.2 Sample Responses](#)
- [4.0 Error Reporting](#)
- [5.0 References](#)

## 1.0 Introduction

A major objective of the Scalable Systems Software [SSS] Center is to create a scalable and modular infrastructure for resource management and accounting on terascale clusters including resource scheduling, grid-scheduling, node daemon support, comprehensive usage accounting and user interfaces emphasizing portability to terascale vendor operating systems. Existing resource management and accounting components feature disparate APIs (Application Programming Interfaces) requiring various forms of application coding to interact with other components.

This document proposes a common message format expressed in an XML request-response syntax to be considered as the foundation of a standard for communications between and among resource management and accounting software components. In this document this standard is expressed in two levels of generality. The features of the core SSSRMAP protocol common to all resource management and accounting components in general are described in the main body of this document. The aspects of the syntax specific to individual components are described in component-specific binding documents.

## 2.0 Conventions Used in This Document

### 2.1 Keywords

The keywords “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC2119 [RFC2119].

### 2.2 XML Case Conventions

In order to enforce a consistent capitalization and naming convention across all SSSRMAP specifications “Upper Camel Case” (UCC) and “Lower Camel Case” (LCC) Capitalization styles shall be used. UCC style capitalizes the first character of each word and compounds the name. LCC style capitalizes the first character of each word except the first word. [XML\_CONV][FED\_XML]

1. SSSRMAP XML Schema and XML instance documents SHALL use the following conventions:
  - Element names SHALL be in UCC convention (example: <UpperCamelCaseElement/>).
  - Attribute names SHALL be in LCC convention (example: <UpperCamelCaseElement lowerCamelCaseAttribute="Whatever"/>).
2. General rules for all names are:
  - Acronyms SHOULD be avoided, but in cases where they are used, the capitalization SHALL remain (example: XMLSignature).
  - Underscores (`_`), periods (`.`) and dashes (`-`) MUST NOT be used (example: use JobId instead of JOB.ID, Job\_ID or job-id).

### 2.3 Schema Definitions

SSSRMAP Schema Definitions appear like this

In case of disagreement between the schema file and this specification, the schema file takes precedence.

## 3.0 Encoding

Encoding tells how a message is represented when exchanged. SSSRMAP data exchange messages SHALL be defined in terms of XML schema [XML\_SCHEMA].

### 3.1 Schema Header and Namespaces

The header of the schema definition is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:sssrmmap="http://scidac.org/ScalableSystems/SSSRMAP"
  targetNamespace="http://www.scidac.org/ScalableSystems/SSSRMAP"
  elementFormDefault="qualified">
```

### 3.2 Element Descriptions

The following subsections describe the elements that make up SSSRMAP messages. SSSRMAP messages are transmitted in the Body and Envelope elements as described in the SSSRMAP Wire Protocol specification [WIRE\_PROTOCOL].

#### *The Request Element*

The `Request` element specifies an individual request. An object-oriented request will have at least one `Object` element while a functional request will not have one. Depending on context, the `Request` element MAY contain one or more `Get` elements or one or more `Set` elements and any number of `Where` elements. `Option`, `Data`, `File` or `Count` elements may also be included. If a component supports it, chunking may be requested where large response data is possible. Setting the chunking attribute to “True” requests that the server break a large response into multiple chunks (each with their own envelope) so they can be processed in separate pieces.

Only an `action` attribute is required. All other attributes are optional.

Attribute	Description
<b>action</b>	Specifies the action or function to be performed
<b>actor</b>	The authenticated user sending the request
<b>id</b>	Uniquely maps the request to the appropriate response
<b>chunking</b>	Requests that segmentation be used for large response data if set to “True”
<b>chunkSize</b>	Requests that the segmentation size be no larger than the specified amount

```

<complexType name="RequestType">
  <choice minOccurs="0" maxOccurs="unbounded">
    <element ref="sssrmmap:Object" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="sssrmmap:Option" minOccurs="0" maxOccurs="unbounded"/>
    <choice minOccurs="0" maxOccurs="1">
      <element ref="sssrmmap:Get" minOccurs="1" maxOccurs="unbounded"/>
      <element ref="sssrmmap:Set" minOccurs="1" maxOccurs="unbounded"/>
    </choice>
    <element ref="sssrmmap:Where" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="sssrmmap:Data" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="sssrmmap:Count" minOccurs="0" maxOccurs="1"/>
    <any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
  </choice>
  <attribute name="action" type="string" use="required"/>
  <attribute name="actor" type="string" use="required"/>
  <attribute name="id" type="string" use="optional"/>
  <attribute name="chunking" type="sssrmmap:BoolType" use="optional"/>
  <attribute name="chunkSize" type="positiveInteger" use="optional"/>
</complexType>

<element name="Request" type="sssrmmap:RequestType"/>

```

### The Object Element

The `Object` element is used in an object-oriented request to specify the object receiving the action. It is possible to have multiple `Object` elements in a request if an implementation supports multi-object queries.

The object class name is specified as text content. All attributes are optional.

- `join` – the type of join to be performed with the preceding object
  - A `join` attribute of “Inner” specifies an inner join. This is the default.
  - A `join` attribute of “FullOuter” specifies a full outer join.
  - A `join` attribute of “LeftOuter” specifies a left outer join.
  - A `join` attribute of “RightOuter” specifies a right outer join.
  - A `join` attribute of “Cross” specifies a cross join.
  - A `join` attribute of “Union” specifies a union join.

```

<complexType name="ObjectType">
  <simpleContent>
    <extension base="string">
      <attribute name="join" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="Object" type="sssrmmap:ObjectType"/>

```

### The Get Element

The `Get` element is used to indicate the data fields to be returned in a query. `Get` is typically used within requests with `action="query"`. Multiple `Get` elements cause the fields to be returned in the order specified. If no `Get` elements are specified, the query will return a default set of fields.

Only a name attribute is required. All other attributes are optional.

Attribute	Description
<b>name</b>	The name of the data field to be returned. This MUST be of the form of a “Modified XPATH expression” as described in a later section.
<b>op</b>	The operator to be used to aggregate or perform an operation on the returned values. <ul style="list-style-type: none"> <li>• An <i>op</i> attribute of “Sort” specifies an ascending sort operation</li> <li>• An <i>op</i> attribute of “Tros” specifies a descending sort operation</li> <li>• An <i>op</i> attribute of “Sum” returns the sum (only valid for numeric values)</li> <li>• An <i>op</i> attribute of “Max” returns the maximum value</li> <li>• An <i>op</i> attribute of “Min” returns the minimum value</li> <li>• An <i>op</i> attribute of “Count” returns the number of values</li> <li>• An <i>op</i> attribute of “Average” returns the average of the values</li> <li>• An <i>op</i> attribute of “GroupBy” signifies that aggregates are grouped by this field</li> </ul>
<b>object</b>	Specifies the object for which you want the named attribute in a multi-object query.
<b>units</b>	The units in which to return the value (if applicable)

```

<complexType name="GetType">
  <attribute name="name" type="string" use="required"/>
  <attribute name="object" type="string" use="optional"/>
  <attribute name="op" type="sssrmapi:GetOperatorType" use="optional"/>
  <attribute name="units" type="string" use="optional"/>
</complexType>

<element name="Get" type="sssrmapi:GetType"/>

<simpleType name="GetOperatorType">
  <restriction base="string">
    <enumeration value="Sort"/>
    <enumeration value="Tros"/>
    <enumeration value="Count"/>
    <enumeration value="Sum"/>
    <enumeration value="Max"/>
    <enumeration value="Min"/>
    <enumeration value="Average"/>
    <enumeration value="GroupBy"/>
  </restriction>
</simpleType>

```

### The Set Element

The Set element is used to specify the object data fields to be assigned values. Set is typically used within requests with `action="Create"` or `action="Modify"`. The use of Get or Set elements within a request is mutually exclusive.

The assignment value (to which the field is being changed) is specified as the text content. A Set element without a value may be used as an assertion flag. Only the `name` attribute is required. All other attributes are optional.

Attribute	Description
<b>name</b>	The name of the field being assigned a value. This MUST be of the form of a “Modified XPATH expression” as described in a later section.
<b>op</b>	The operator to be used in assigning a new value to the name. If an <code>op</code> attribute is not specified and a value is specified, the specified value will be assigned to the named field (“assign”). <ul style="list-style-type: none"> <li>• An <code>op</code> attribute of “Assign” assigns value to the named field</li> <li>• An <code>op</code> attribute of “Inc” increments the named field by the value</li> <li>• An <code>op</code> attribute of “Dec” decrements the named field by the value</li> </ul>
<b>units</b>	The units corresponding to the value being set

```

<complexType name="SetType">
  <simpleContent>
    <extension base="string">
      <attribute name="name" type="string" use="required"/>
      <attribute name="op" type="sssrmap:SetOperatorType" use="optional"/>
      <attribute name="units" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="Set" type="sssrmap:SetType"/>

<simpleType name="SetOperatorType">
  <restriction base="string">
    <enumeration value="Assign"/>
    <enumeration value="Inc"/>
    <enumeration value="Dec"/>
  </restriction>
</simpleType>

```

### The Where Element

A `Request` element may contain one or more `Where` elements that specify the search conditions for which objects the action is to be performed on.

The condition value (against which the field is tested) is specified as the text content. A `Where` element without a value may be used as a truth test. Only the `name` attribute is required. All other attributes are optional.

Attribute	Description
<b>name</b>	The name of the data field to be tested. This MUST be of the form of a “Modified XPATH expression” as described in a later section.

Attribute	Description
<b>op</b>	<p>The operator to be used to test the name against the value. If an <code>op</code> attribute is not specified and a value is specified, the field will be tested whether it is equal to the value ("EQ").</p> <ul style="list-style-type: none"> <li>• An <code>op</code> attribute of "EQ" specifies an equality comparison</li> <li>• An <code>op</code> attribute of "LT" specifies a "less than" comparison</li> <li>• An <code>op</code> attribute of "GT" specifies a "greater than" comparison</li> <li>• An <code>op</code> attribute of "LE" specifies a "less than or equal to" test</li> <li>• An <code>op</code> attribute of "GE" specifies a "greater than or equal to" test</li> <li>• An <code>op</code> attribute of "NE" specifies a "not equal to" test</li> <li>• An <code>op</code> attribute of "Match" specifies a regular expression matching comparison</li> </ul>
<b>conj</b>	<p>Indicates whether this test is to be anded or ored with the immediately preceding where condition</p> <ul style="list-style-type: none"> <li>• A <code>conj</code> attribute of "And" specifies an "and" conjunction</li> <li>• A <code>conj</code> attribute of "Or" specifies an "or" condition</li> <li>• A <code>conj</code> attribute of "AndNot" specifies an "and not" conjunction</li> <li>• A <code>conj</code> attribute of "OrNot" specifies an "or not" condition</li> </ul>
<b>group</b>	<p>Indicates an increase or decrease of parentheses grouping depth</p> <ul style="list-style-type: none"> <li>• A positive number indicates the number of left parentheses to precede the condition, i.e. <code>group="2"</code> represents "((condition)".</li> <li>• A negative number indicates the number of right parentheses to follow the condition, i.e. <code>group="-2"</code> represents "condition))".</li> </ul>
<b>object</b>	Specifies the object for the first operand in a multi-object query.
<b>subject</b>	Specifies the object for the second operand in a multi-object query.
<b>units</b>	Indicates the units to be used in the value comparison

```

<complexType name="WhereType">
  <simpleContent>
    <extension base="string">
      <attribute name="name" type="string" use="required"/>
      <attribute name="op" type="sssrmop:OperatorType" use="optional"/>
      <attribute name="conj" type="sssrmop:ConjunctionType" use="optional"/>
      <attribute name="group" type="integer" use="optional"/>
      <attribute name="units" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="Where" type="sssrmop:WhereType"/>

<simpleType name="WhereOperatorType">
  <restriction base="string">
    <enumeration value="EQ"/>
    <enumeration value="GT"/>
    <enumeration value="LT"/>
    <enumeration value="GE"/>
    <enumeration value="LE"/>
    <enumeration value="NE"/>
    <enumeration value="Match"/>
  </restriction>
</simpleType>

```

### The Option Element

The `Option` element is used to indicate processing options for the command. An option might be used to indicate that command usage or special formatting is desired, or that the command is to be invoked with particular options.

The option value is specified as the text content. An `Option` element without a value may be used as an assertion flag. Only the `name` attribute is required. All other attributes are optional.

Attribute	Description
<b>name</b>	The name of the field being assigned a value
<b>op</b>	The operator to be used to disassert the option <ul style="list-style-type: none"> <li>An <code>op</code> attribute of "Not" specifies that the option is not asserted</li> </ul>
<b>conj</b>	Indicates whether this test is to be anded or ored with the immediately preceding where condition <ul style="list-style-type: none"> <li>A <code>conj</code> attribute of "And" specifies an "and" conjunction</li> <li>A <code>conj</code> attribute of "Or" specifies an "or" condition</li> <li>A <code>conj</code> attribute of "AndNot" specifies an "and not" conjunction</li> <li>A <code>conj</code> attribute of "OrNot" specifies an "or not" condition</li> </ul>

```

<complexType name="OptionType">
  <simpleContent>
    <extension base="string">
      <attribute name="name" type="string" use="required"/>
      <attribute name="op" type="sssrmap:OptionOperatorType" use="optional"/>
      <attribute name="conj" type="sssrmap:ConjunctionType" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="Option" type="sssrmap:OptionType"/>

<simpleType name="OptionOperatorType">
  <restriction base="string">
    <enumeration value="Not"/>
  </restriction>
</simpleType>

```

### The Data Element

A Request or Response element may have one or more Data elements that allow the supplying of context-specific data. A request might pass in a structured object via a Data element to be acted upon. Typically a query will result in a response with the data encapsulated within a Data element.

The following attributes are optional:

Attribute	Description
<b>name</b>	Object name describing the contents of the data
<b>type</b>	Describing the form in which the data is represented <ul style="list-style-type: none"> <li>• A <code>type</code> attribute of "XML" indicates the data has internal xml structure and can be recursively parsed by an XML parser</li> <li>• A <code>type</code> attribute of "Binary" indicates the data is an opaque dataset consisting of binary data</li> <li>• A <code>type</code> attribute of "String" indicates the data is an ASCII string</li> <li>• A <code>type</code> attribute of "Int" indicates the data is an integer</li> <li>• A <code>type</code> attribute of "Text" indicates the data is in formatted human-readable text</li> <li>• A <code>type</code> attribute of "HTML" indicates the data is represented in HTML</li> </ul>

```

<complexType name="DataType">
  <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="name" type="string" use="optional"/>
  <attribute ref="sssrmap:Type" use="optional"/>
</complexType>

<element name="data" type="sssrmap:DataType"/>

```

### The File Element

A Request or Response element may have one or more File elements of type String that allow the inclusion of files. The files may be either text or binary and may be referenced by objects inside the Data element. A file may be compressed using the gzip algorithm [ZIP]. A binary file or a compressed file must

be base64 encoded as defined in XML Digital Signatures (“<http://www.w3.org/2000/09/xmlldsig#base64>”). Metadata describing the modes and properties of the resulting file are passed as parameters. The text or base64 encoded file data forms the string content of the `File` element.

The following attributes are optional:

Attribute	Description
<b>id</b>	Specifies an identifier that allows the file to be referenced from within another object. If more than one <code>File</code> elements are specified, this attribute is REQUIRED in each of them.
<b>name</b>	Specifies the name to give the file upon creation on the target system. This can be an absolute or relative pathname (relative to the <code>InitialWorkingDirectory</code> ).
<b>owner</b>	Indicates what owner the file should be changed to. By default it will be changed to the <code>UserId</code> that the authenticated actor maps to on the target system. Note that this function should succeed only if the requestor has the privileges to do so (i.e. authenticated as root).
<b>group</b>	Indicates what group the file should be changed to. By default it will be set to the primary groupid of the <code>UserId</code> that the authenticated actor maps to on the target system. Note that this function should succeed only if the requestor has the proper privileges.
<b>mode</b>	Indicates the permissions the file should possess. By default it will be set according to the default <code>umask</code> for the <code>UserId</code> that the authenticated actor maps to on the target system. Note that this function should not set permissions for the file that exceed the privileges for the actor. These permissions can be specified using either an octal number or symbolic operations (as accepted by the GNU <code>chmod(1)</code> command).
<b>compressed</b>	Indicates whether the file has been compressed <ul style="list-style-type: none"> <li>• A <code>compressed</code> attribute of “True” indicates the file has been compressed.</li> <li>• A <code>compressed</code> attribute of “False” indicates the file has not been compressed. This is the default.</li> </ul>
<b>encoded</b>	Indicates whether the file has been base64 encoded <ul style="list-style-type: none"> <li>• An <code>encoded</code> attribute of “True” indicates the file has been encoded.</li> <li>• An <code>encoded</code> attribute of “False” indicates the file has not been encoded. This is the default.</li> </ul>

```

<complexType name="FileType">
  <sequence>
    <any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </sequence>
  <attribute name="name" type="string" use="optional"/>
  <attribute name="owner" type="string" use="optional"/>
  <attribute name="group" type="string" use="optional"/>
  <attribute name="mode" type="string" use="optional"/>
  <attribute name="compressed" type="boolean" use="optional"/>
  <attribute name="encoded" type="boolean" use="optional"/>
</complexType>

<element name="file" type="sssrmap:FileType"/>

```

### The Count Element

A single `Count` element may be included within a `Request` or `Response` and is context-specific. This can be used to represent the number of objects acted upon or returned.

```
<element name="Count" type="positiveInteger"/>
```

### The Response Element

The `Response` element specifies an individual response. It **MUST** contain a `Status` element. It **MAY** also contain `Count` and any number of `Data` or `File` elements. If chunking has been requested and is supported by the server, a large response may be broken up into multiple chunks (each with their own envelope). The `chunkNum` attribute can be used to indicate which chunk the current one is. The `chunkMax` attribute can be used to determine when all the chunks have been received (all chunks have been received if `chunkNum=chunkMax` or `chunkMax=0`).

It **MAY** have any of the following attributes:

Attribute	Description
<b>id</b>	Uniquely maps the response to the corresponding request
<b>chunkNum</b>	Integer indicating the current chunk number [1 is implied when this attribute is missing or blank]
<b>chunkMax</b>	Integer indicating the number of chunks expected [-1 means unknown but more chunks to follow; 0 means unknown but this is the last chunk; 0 is implied if this attribute is missing or blank]

```

<complexType name="ResponseType">
  <choice minOccurs="0" maxOccurs="unbounded">
    <element ref="sssrmap:Status" minOccurs="1" maxOccurs="1"/>
    <element ref="sssrmap:Count" minOccurs="0" maxOccurs="1"/>
    <element ref="sssrmap:Data" minOccurs="0" maxOccurs="unbounded"/>
    <element ref="sssrmap:File" minOccurs="0" maxOccurs="unbounded"/>
    <any minOccurs="0" maxOccurs="unbounded" namespace="##other"/>
  </choice>
  <attribute name="object" type="string" use="optional"/>
  <attribute name="action" type="string" use="optional"/>
  <attribute name="id" type="string" use="optional"/>
  <attribute name="chunkNum" type="integer" use="optional"/>
  <attribute name="chunkMax" type="integer" use="optional"/>
</complexType>

<element name="Response" type="sssrmap:ResponseType"/>

```

### The Status Element

A Response element **MUST** contain a single Status element that indicates whether the reply represents a success, warning or failure. This element is composed of the child elements Value, Code and Message. Of these, Value and Code are required, and Message is optional.

```
<complexType name="StatusType">
  <choice minOccurs="1" maxOccurs="unbounded">
    <element ref="sssrmmap:Value" minOccurs="1" maxOccurs="1"/>
    <element ref="sssrmmap:Code" minOccurs="1" maxOccurs="1"/>
    <element ref="sssrmmap:Message" minOccurs="0" maxOccurs="1"/>
    <any minOccurs="0" maxOccurs="unbounded" namespace="##other"/>
  </choice>
</complexType>

<element name="Status" type="sssrmmap:StatusType"/>
```

### The Value Element

The Value element is of type String and **MUST** have a value of "Success", "Warning" or "Failure".

```
<simpleType name="StatusValueType">
  <restriction base="string">
    <enumeration value="Success"/>
    <enumeration value="Warning"/>
    <enumeration value="Failure"/>
  </restriction>
</simpleType>

<element name="Value" type="sssrmmap:StatusValueType"/>
```

### The Code Element

A Response element must contain a single Code element that specifies the 3-digit status code for the response. Refer to the next section on Error Reporting for a description and listing of supported status codes.

```
<simpleType name="CodeType">
  <restriction base="string">
    <pattern value="[0-9]{3}"/>
  </restriction>
</simpleType>

<element name="Code" type="sssrmmap:CodeType"/>
```

### The Message Element

A Response element may contain a single Message element that is context specific to the success or failure response. The message should be an error message if status is false. If present for a successful response, it may be used as a human readable message for a user interface.

```
<element name="Message" type="string"/>
```

## 3.3 Modified XPATH Expressions

The name attribute used within the Get, Set and Where Elements **MUST** have the form of a modified XPATH expression as defined in this section. Usually this will just be the simple name of the object property. Some complex objects, such as the SSS Job Object and the SSS Node Object, however, are

represented in a structured way with nested elements. In order to define a consistent and flexible way to access and manipulate these objects as well as keeping the flat XML objects simple and straightforward, SSSRMAP specifies that a “Modified XPATH” syntax be used.

In essence, “Modified XPATH” is defined to be an XPATH [XPATH] expression with the exception that the “//” may be omitted from the beginning of the expression when a document search is desired. Thus, on the server side, a standard XPATH routine can be used by prepending “//” to any expression that does not begin with a “/”.

The response data should always include all of the structure of the queried object necessary to place the requested data in its proper context.

See the XPATH specification for a full description of XPATH. The XPath 1.0 Recommendation is <http://www.w3.org/TR/1999/REC-xpath-19991116>. The [latest version of XPath 1.0](#) is available at <http://www.w3.org/TR/xpath>.

### Sample Modified XPATH Expressions

Consider the following hypothetical object(s) (which might be returned within a Data element).

```
<Job>
  <JobId>PBS.1234.0</JobId>
  <Requested>
    <Memory op="GE">512</Memory>
    <Processors>2</Processors>
    <WallDuration>P3600S</WallDuration>
  </Requested>
  <Utilized>
    <Memory metric="Average">488</Memory>
    <WallDuration>P1441S</WallDuration>
  </Utilized>
</Job>
```

To get everything above for this job you would not need a Get element:

```
<Request action="Query">
  <Object>Job</Object>
  <Where name="JobId">PBS.1234.0</Where>
</Request>
```

If you used `<Get name="JobId"/>` you would get back:

```
<Job>
  <JobId>PBS.1234.0</JobId>
</Job>
```

If you used `<Get name="Memory"/>` (or `name="/Job/*/Memory"`) you would get:

```
<Job>
  <Requested>
    <Memory op="GE">512</Memory>
  </Requested>
  <Utilized>
    <Memory metric="Average">488</Memory>
  </Utilized>
</Job>
```

If you used `<Get name="Requested/Memory"/>` (or `name="/Job/Requested/Memory"`) you would get:

```
<Job>
  <Requested>
    <Memory op="GE">512</Memory>
  </Requested>
</Job>
```

If you used `<Get name="Memory[@metric='Average']"/>` (or `name="Memory[@metric]"`) you would get:

```
<Job>
  <Utilized>
    <Memory metric="Average">488</Memory>
  </Utilized>
</Job>
```

### 3.4 Examples

#### Sample Requests

Requesting a list of nodes with a certain configured memory threshold (batch format):

```
<Request action="Query" id="1">
  <Object>Node</Object>
  <Get name="Name" />
  <Get name="Configured/Memory" />
  <Where name="Configured/Memory" op="GE" units="MB">512</Where>
</Request>
```

Activating a couple of users:

```
<Request action="Modify">
  <Object>User</Object>
  <Set name="Active">True</Set>
  <Where name="Name">scott</Where>
  <Where name="Name" conj="Or"/>brett</Where>
</Request>
```

Submitting a simple job:

```
<Request action="Submit">
  <Object>Job</Object>
  <Data>
    <Job>
      <User>xdp</User>
      <Account>youraccount</Account>
      <Command>myprogram</Command>
      <InitialWorkingDirectory>/usr/home/scl/xdp</InitialWorkingDirectory>
      <RequestedNodes>4</RequestedNodes>
      <RequestedWCTime>100</RequestedWCTime>
    </Job>
  </Data>
</Request>
```

## Sample Responses

A response to the available memory nodes query (batch format)

```
<Response id="1">
  <Status>
    <Value>Success</Value>
    <Code>000</Code>
  </Status>
  <Count>2</Count>
  <Data>
    <Node>
      <Name>fr01n01</Name>
      <Configured>
        <Memory>512</Memory>
      </Configured>
    </Node>
    <Node>
      <Name>fr12n04</Name>
      <Configured>
        <Memory>1024</Memory>
      </Configured>
    </Node>
  </Data>
</Response>
```

Two users successfully activated

```
<Response>
  <Status>
    <Code>000</Code>
    <Message>Two users were successfully modified</Message>
  </Status>
  <Count>2</Count>
</Response>
```

A failed job submission:

```
<Response>
  <Status>
    <Value>Failure</Value>
    <Code>711</Code>
    <Message>Invalid account specified. The job was not submitted.</Message>
  </Status>
</Response>
```

## 4.0 Error Reporting

SSSRMAP requests will return a status and a 3-digit response code to signify success or failure conditions. When a request is successful, a corresponding response is returned with the `status` element set to `Success` and the `code` element set to "000". When a request results in an error detected by the server, a response is returned with the `status` element set to `Failure` and a 3-digit error code in the `code` element. An optional human-readable message may also be included in a failure response providing context-specific detail about the failure. The default message language is US English. (The status flag makes it easy to signal success or failure and allows the receiving peer some freedom in the amount of parsing it wants to do on failure [BXXP]).

**Success codes:**

Code	Response Text in US English
<b>0xx</b>	Request was successful
<b>000</b>	General Success
<b>010</b>	Help/usage reply
<b>020</b>	Status reply
<b>030</b>	Subscription successful
<b>035</b>	Notification successful (Ack)
<b>040</b>	Registration successful
<b>050-079</b>	Component-defined
<b>080-099</b>	Application-defined

**Warning codes:**

Code	Response Text in US English
<b>1xx</b>	Request was successful but includes a warning
<b>100</b>	General warning (examine message for details)
<b>102</b>	Check result (Did what you asked but may not have been what you intended -- or information is suspect)
<b>110</b>	Wire Protocol or Network warning
<b>112</b>	Redirect
<b>114</b>	Protocol warning (something was wrong with the protocol, but best effort guesses were applied to fulfill the request)
<b>120</b>	Message Format warning

Code	Response Text in US English
122	Incomplete specification (request missing some essential information -- best effort guess applied)
124	Format warning (something was wrong with the format but best effort guesses were applied to fulfill the request)
130	Security warning
132	Insecure request
134	Insufficient privileges (Response was sanitized or reduced in scope due to lack of privileges)
140	Content or action warning
142	No content (The server has processed the request but there is no data to be returned)
144	No action taken (nothing acted upon -- i.e. deletion request did not match any objects)
146	Partial content
148	Partial action taken
150-179	Component-defined
180-199	Application-defined

### Wire protocol codes:

Code	Response Text in US English
2xx	A problem occurred in the wire protocol or network
200	General wire protocol or network error
210	Network failure
212	Cannot resolve host name

Code	Response Text in US English
214	Cannot resolve service port
216	Cannot create socket
218	Cannot bind socket
220	Connection failure
222	Cannot connect
224	Cannot send data
226	Cannot receive data
230	Connection rejected
232	Timed out
234	Too busy
236	Message too large
240	Framing failure
242	Malformed framing protocol
244	Invalid payload size
246	Unexpected end of file
250-279	Component-defined
280-299	Application-defined

**Message format codes:**

Code	Response Text in US English
<b>3xx</b>	A problem occurred in the message format
<b>300</b>	General message format error
<b>302</b>	Malformed XML document
<b>304</b>	Validation error(XML Schema)
<b>306</b>	Namespace error
<b>308</b>	Invalid message type (Something other than Request or Response in Body
<b>310</b>	General syntax error in request
<b>311</b>	Object incorrectly (or not) specified
<b>312</b>	Action incorrectly (or not) specified
<b>313</b>	Invalid Action
<b>314</b>	Missing required element or attribute
<b>315</b>	Invalid Object (or Object-Action combination
<b>316</b>	Invalid element or attribute name
<b>317</b>	Illegal value for element or attribute
<b>318</b>	Illegal combination
<b>319</b>	Malformed Data
<b>320</b>	General syntax error in response
<b>321</b>	Status incorrectly (or not)specified
<b>322</b>	Code incorrectly (or not)specified

Code	Response Text in US English
324	Missing required element or attribute
326	Invalid element or attribute name
327	Illegal value for element or attribute
328	Illegal combination
329	Malformed Data
340	Pipelining failure
342	Request identifier is not unique
344	Multiple messages not supported
346	Mixed messages not supported (Both requests and responses in same batch)
348	Request/response count mismatch
350-379	Component-defined
380-399	Application-defined

**Security codes:**

Code	Response Text in US English
4xx	A security requirement was not fulfilled
400	General security error
410	Negotiation failure
412	Not understood
414	Not supported

Code	Response Text in US English
416	Not accepted
420	Authentication failure
422	Signature failed at client
424	Authentication failed at server
426	Signature failed at server
428	Authentication failed at client
430	Encryption failure
432	Encryption failed at client
434	Decryption failed at server
436	Encryption failed at server
438	Decryption failed at client
440	Authorization failure
442	Authorization failed at client
444	Authorization failed at server
450-479	Component-defined
480-499	Application-defined

**Event management codes:**

Code	Response Text in US English
5xx	Failure conditions in event messaging

Code	Response Text in US English
500	General Event Management failure
510	Subscription failed
520	Notification failed
550-579	Component-defined
580-599	Application-defined

**Reserved codes:**

Code	Response Text in US English
6xx	Reserved for future use

**Server application codes:**

Code	Response Text in US English
7xx	A server-side application-specific error occurred
700	General failure
710	Not supported
712	Not understood
720	Internal error
730	Resource unavailable (insufficient resources -- software, hardware or a service I rely upon is down)
740	Business logic
750-779	Component-defined
780-799	Application-defined

**Client application codes:**

Code	Response Text in US English
<b>8xx</b>	A client-side application-specific error occurred
<b>800</b>	General failure
<b>810</b>	Not supported
<b>812</b>	Not understood
<b>820</b>	Internal error
<b>830</b>	Resource unavailable
<b>840</b>	Business logic
<b>850-879</b>	Component-defined
<b>880-899</b>	Application-defined

**Miscellaneous codes:**

Code	Response Text in US English
<b>9xx</b>	Miscellaneous failures
<b>999</b>	Unknown failure

**5.0 References**

[BEEP] M. Rose, “The Blocks Extensible Exchange Protocol Core”, [RFC 3080](#), March 2001.

[FED\_XML] “[U.S. Federal XML Guidelines](#)”.

[HMAC] H. Krawczyk, M. Bellare, R. Canetti, “HMAC, Keyed-Hashing for Message Authentication”, [RFC 2104](#), February 1997.

[HTTP] “Hypertext Transfer Protocol – HTTP/1.1”, [RFC 2616](#), June 1999.

[RFC2119] S. Bradner, “Key Words for Use in RFCs to Indicate Requirement Levels”, [RFC 2119](#), March 1997.

[RFC3117] M. Rose, “On the Design of Application Protocols”, [Informational RFC 3117](#), November 2001.

[SHA-1] U.S. Department of Commerce/National Institute of Standards and Technology, "[Secure Hash Standard](#)", FIPS PUB 180-1.

[SSS] "Scalable Systems Software", <http://www.scidac.org/ScalableSystems>

[WIRE\_PROTOCOL] S. Jackson, B. Bode, D. Jackson, K. Walker, "Systems Software Resource Management and Accounting Protocol (SSSRMAP) Wire Protocol", [SSS Resource Management and Accounting Documents](#), January 2004.

[XML] Bray, T., et al, "[Extensible Markup Language \(XML\) 1.0 \(Second Edition\)](#)", 6 October 2000.

[XML\_CONV] "[I-X and <I-N-CA> XML Conventions](#)".

[XML\_DSIG] D. Eastlake, J. Reagle Jr., D. Solo, "[XML Signature Syntax and Processing](#)", W3C Recommendation, 12 February 2002.

[XML\_ENC] T. Imamura, B. Dillaway, E. Smon, "[XML Encryption Syntax and Processing](#)", W3C Candidate Recommendation, 4 March 2002.

[XML\_SCHEMA] D. Beech, M. Maloney, N. Mendelshohn, "[XML Schema Part 1: Structures Working Draft](#)", April 2000.

[XPath 1.0] J. Clark, S. DeRose, "[XML Path Language \(XPath\) Version 1.0](#)", 16 November 1999.

[XRP] E. Brunner-Williams, A. Damaraju, N. Zhang, "[Extensible Registry Protocol \(XRP\)](#)", Internet Draft, expired August 2001.

[ZIP] J. Gailly, M. Adler, "The gzip home page", <http://www.gzip.org/>

## Scalable Systems Software Node Object Specification

SSS Node Object Specification  
Release Version 3.1.0  
26 April 2011

Scott Jackson, PNNL  
David Jackson, Ames Lab  
Brett Bode, Ames Lab

### Status of This Memo

This is a specification of the node object to be used by Scalable Systems Software compliant components. It is envisioned for this specification to be used in conjunction with the SSSRMAP protocol with the node object passed in the Data field of Requests and Responses. Queries can be issued to a node-cognizant component in the form of modified XPATH expressions to the Get field to extract specific information from the node object as described in the SSSRMAP protocol.

### Abstract

This document describes the syntax and structure of the SSS node object. This node model takes into account various node property categories such as whether it represents a configured, available or utilized property.

## Table of Contents

- [Scalable Systems Software Node Object Specification](#)
- [Table of Contents](#)
- [1.0 Introduction](#)
  - [1.1 Goals](#)
  - [1.2 Examples](#)
    - [1.2.1 Simple Example](#)
    - [1.2.2 Elaborate Example](#)
- [2.0 Conventions Used in This Document](#)
  - [2.1 Keywords](#)
  - [2.2 Table Column Interpretations](#)
  - [2.3 Element Syntax Cardinality](#)
- [3.0 The Node Model](#)
- [4.0 Node Element](#)
  - [4.1 Uncategorized Node Properties](#)
    - [4.1.1 Simple Node Properties](#)
    - [4.1.2 Extension Element](#)
  - [4.2 Property Categories](#)
    - [4.2.1 Configured Element](#)
    - [4.2.2 Available Element](#)
    - [4.2.3 Utilized Element](#)
  - [4.3 Categorized Node Properties](#)
    - [4.3.1 Consumable Resources](#)
    - [4.3.2 Resource Element](#)
- [Appendix A](#)
- [Units of Measure Abbreviations](#)

## 1.0 Introduction

This specification proposes a standard XML representation for a node object for use by the various components in the SSS Resource Management System. This object will be used in multiple contexts and by multiple components. It is anticipated that this object will be passed via the Data Element of SSSRMAP Requests and Responses.

## 1.1 Goals

There are several goals motivating the design of this representation.

It needs to be inherently flexible. We recognize we will not be able to exhaustively include the ever-changing node properties and capabilities that constantly arise.

The same node object should be used at all stages of its lifecycle. This object needs to distinguish between configured, available and utilized properties of a node.

Its design takes into account the properties and structure required to function in a meta or grid environment. It should eventually include the capability of resolving namespace and locality issues, though the earliest versions will ignore this requirement.

One should not have to make multiple queries to obtain a single piece of information — i.e. there should not be two mutually exclusive ways to represent a node resource.

It needs to support resource metric as well as unit specifications.

## 1.2 Examples

### Simple Example

This example shows a simple expression of the Node object.

```
<Node>
  <Id>Node64</Id>
  <Configured>
    <Processors>2</Processors>
    <Memory>512</Memory>
  </Configured>
</Node>
```

### Elaborate Example

This example shows a more elaborate Node object.

```
<Node>
  <Id>64</Id>
  <Name>Netpipe2</Name>
  <Feature>BigMem</Feature>
  <Feature>NetOC12</Feature>
  <Opsys>AIX</Opsys>
  <Arch>Power4</Arch>
  <Configured>
    <Processors>16</Processors>
    <Memory units="MB">512</Memory>
    <Swap>512</Swap>
  </Configured>
  <Available>
    <Processors>7</Processors>
    <Memory metric="Instantaneous">143</Memory>
  </Available>
  <Utilized>
    <Processors wallDuration="3576">8</Processors>
    <Memory metric="Average" wallDuration="3576">400</Memory>
  </Utilized>
</Node>
```

## 2.0 Conventions Used in This Document

### 2.1 Keywords

The keywords **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted as described in [RFC2119](#).

### 2.2 Table Column Interpretations

In the property tables, the columns are interpreted to have the following meanings:

Property	Description
<b>Element Name</b>	Name of the XML element (xsd:element)
<b>Type</b>	Data type defined by xsd (XML Schema Definition) as: <ul style="list-style-type: none"> <li>• String — xsd:string(a finite length sequence of printable characters)</li> <li>• Integer — xsd:integer(a signed finite length sequence of decimal digits)</li> <li>• Float — xsd:float (single-precision 32-bit floating point)</li> <li>• Boolean — xsd:boolean (consists of the literals “true” or “false”)</li> <li>• DateTime — xsd:dateTime (discreet time values are represented in ISO 8601 extended format CCYY-MM-DDThh:mm:ss where CC represents the century, YY the year, MM the month and DD the day. The letter T is the date/time separator and hh, mm, ss represent hour, minute and second respectively. This representation may be immediately followed by a Z to indicate Coordinated Universal Time (UTC) or, to indicate the time zone, i.e. the difference between the local time and Coordinated Universal Time, immediately followed by a sign, + or -, followed by the difference from UTC.)</li> <li>• Duration — xsd:duration (a duration of time is represented in ISO 8601 extended format PnYnMnDtnHnMnS, where nY represents the number of years, nM the number of months, nD the number of days, T is the date/time separator, nH the number of hours, nM the number of minutes and nS the number of seconds. The number of seconds can include decimal digits to arbitrary precision.)</li> </ul>
<b>Description</b>	Brief description of the meaning of the property
<b>Appearance</b>	This column indicates whether the given property has to appear within the parent element. It assumes the following meanings: <ul style="list-style-type: none"> <li>• <b>MUST</b> — This property is <b>REQUIRED</b> when the parent is specified.</li> <li>• <b>SHOULD</b> — A compliant implementation <b>SHOULD</b> support this property.</li> <li>• <b>MAY</b> — A compliant implementation <b>MAY</b> support this property.</li> </ul>

Property	Description
<b>Compliance</b>	<p>The column indicates whether a compliant implementation has to support the given property.</p> <ul style="list-style-type: none"> <li>• <b>MUST</b> — A compliant implementation <b>MUST</b> support this property.</li> <li>• <b>SHOULD</b> — A compliant implementation <b>SHOULD</b> support this property.</li> <li>• <b>MAY</b> — A compliant implementation <b>MAY</b> support this property.</li> </ul>
<b>Categories</b>	<p>Some properties may be categorized into one of several categories. Letters in this column indicate that the given property can be classified in the following property categories.</p> <ul style="list-style-type: none"> <li>• <b>C</b> — This property can be encompassed in a Configured element.</li> <li>• <b>A</b> — This property can be encompassed in an Available element.</li> <li>• <b>U</b> — This property can be encompassed in a Utilized element.</li> </ul>

### 2.3 Element Syntax Cardinality

The cardinality of elements in the element syntax sections may make use of regular expression wildcards with the following meanings:

Wildcard	Description
*	Zero or more occurrences
+	One or more occurrences
?	Zero or one occurrences

The absence of one of these symbols implies one and only one occurrence.

### 3.0 The Node Model

The primary element within the node model is a node. One can speak of some node properties as being a configured, available or utilized property of the node.

### 4.0 Node Element

The Node element is the root element of a node object and is used to encapsulate a node.

- A node object **MUST** have exactly one Node element.
- A compliant implementation **MUST** support this element.
- A node **MUST** specify one or more Node Properties.

## 4.1 Uncategorized Node Properties

Uncategorized Node Properties are properties that apply to the node as a whole and do not need to be distinguished between being configured, available or utilized. These include the node id and other optional node properties.

### Simple Node Properties

Simple (unstructured) node properties are enumerated in the table below.

Table 3-13: Simple Node Properties

Element Name	Type	Description	Appearance	Compliance
<b>Id</b>	String	Node identifier	MUST	MUST
<b>Name</b>	String	Node name or pattern	MAY	MAY
<b>OpSys</b>	String	Operating System	MAY	SHOULD
<b>Arch</b>	String	Architecture	MAY	SHOULD
<b>Description</b>	String	Description of the node	MAY	MAY
<b>State</b>	String	State of the node. Valid states may include Offline, Configured, Unknown, Idle, and Busy.	SHOULD	MUST
<b>Features</b>	String	Arbitrary named features of the node (comma-delimited string)	MAY	SHOULD

### Extension Element

The `Extension` element provides a means to pass extensible properties with the node object. Some applications may find it easier to deal with a named extension property than discover and handle elements for which they do not understand or anticipate by name.

- A compliant implementation **MAY** support this element.
- This element **MUST** have a name attribute that is of type `String` and represents the name of the extension property. A compliant implementation **MUST** support this attribute if this element is supported.
- This element **MAY** have a type attribute that is of type `String` and provides a hint about the context within which the property should be understood. A compliant implementation **SHOULD** support this attribute if this element is supported.
- The character content of this element is of type `String` and is the value of the extension property.

The following is an example of an `Extension` element:

```
<Extension type="Chemistry" name="Software">NWChem</Extension>
```

## 4.2 Property Categories

Certain node properties (particularly consumable resources) need to be classified as being in a particular category. This is done when it is necessary to distinguish between a property that is configured versus a property that is available or utilized. For example, a node might be configured with 16 processors. At a particular time, 8 might be utilized, 7 might be available and 1 disabled. When a node property must be categorized to be understood properly, the property **MUST** be enveloped within the appropriate Property Category Element.

### *Configured Element*

A configured node property reflects resources pertaining to the node that could in principle be used though they may not be available at this time. This information could be used to determine if a job could ever conceivably run on a given node.

- A compliant implementation **MUST** support this element.

The following is an example of using Configured Properties:

```
<Configured>
  <Processors>16</Processors>
  <Memory units="MB">512</Memory>
</Configured>
```

### *Available Element*

An available node property refers to a resource that is currently available for use.

- A compliant implementation **SHOULD** support this element.

The following is an example of specifying available properties:

```
<Available>
  <Processors>7</Processors>
  <Memory units="MB">256</Memory>
</Available>
```

### *Utilized Element*

A utilized node property reflects resources that are currently utilized.

- A compliant implementation **SHOULD** support this element.

The following is an example of specifying utilized properties:

```
<Utilized>
  <Processors>8</Processors>
  <Memory metric="Average">207</Memory>
</Utilized>
```

### 4.3 Categorized Node Properties

#### Consumable Resources

Consumable Resources are a special group of node properties that can have additional attributes and can be used in multiple categories. In general a consumable resource is a resource that can be consumed in a measurable quantity.

- A consumable resource **MUST** be categorized as being a configured, available or utilized node property by being a child element of a Configured, Available or Utilized element respectively.
- A consumable resource **MAY** have a units attribute that is of type String that specifies the units by which it is being measured. If this attribute is omitted, a default unit is implied. A compliant implementation **MAY** support this attribute if the element is supported.
- A consumable resource **MAY** have a metric attribute that is of type String that specifies the type of measurement being described. For example, the measurement may be a Total, an Average, a Min or a Max. A compliant implementation **MAY** support this attribute if the element is supported.
- A consumable resource **MAY** have a wallDuration attribute of type Duration that indicates the amount of time for which that resource was used. This need only be specified if the resource was used for a different amount of time than the wallDuration for the step. A compliant implementation **MAY** support this attribute if the element is supported.
- A consumable resource **MAY** have a consumptionRate attribute of type Float that indicates the average percentage that a resource was used over its wallDuration. For example, an overbooked SMP running 100 jobs across 32 processors may wish to scale the usage and charge by the average fraction of processor usage actually delivered. A compliant implementation **MAY** support this attribute if the element is supported.

A list of simple consumable resources is listed in the table below.

Table 3-14: Consumable Resource Node Properties

Element Name	Type	Description	Appearance	Compliance	Categories
<b>Processors</b>	Integer	Number of processors	MAY	MUST	CAU
<b>Memory</b>	Float	Amount of memory	MAY	SHOULD	CAU
<b>Disk</b>	Float	Amount of disk	MAY	SHOULD	CAU
<b>Swap</b>	Float	Amount of virtual memory	MAY	MAY	CAU
<b>Network</b>	Float	Amount of network	MAY	MAY	CAU

The following are two examples for specifying a consumable resource:

```
<Memory metric="Max" units="GB">483</Memory>
<Processors wallDuration="1234" consumptionRate="0.63">4</Processors>
```

### Resource Element

In addition to the consumable resources enumerated in the above table, an extensible consumable resource is defined by the Resource element.

- A compliant implementation SHOULD support this element.
- This element MAY appear zero or more times within a given set of node properties.
- Like the other consumable resources, this property MUST be categorized as a configured, available or utilized property by being encompassed in the appropriate elements.
- This element is of type Float.
- It shares the other same properties and attributes as the other consumable resources but it requires an additional name (and optional type) attribute to describe it.
- This element MUST have a name attribute of type String that indicates the type of consumable resource being measured. A compliant implementation MUST support this attribute if the element is supported.
- This element MAY have a type attribute of type String that distinguishes it within a general resource class. A compliant implementation SHOULD support this attribute if the element is supported.

The following are two examples for specifying a Resource element:

```
<Resource name="License" type="MATLAB">1</Resource>
<Resource name="Telescope" type="Zoom2000" wallDuration="750"
metric="KX">10</Resource>
```

## 4.4 Node Reference

When a simple reference to a predefined node is needed in an encapsulating element, a Node element is used with the text content being the node id:

```
<Node>node1</Node>
```

- This element MAY have an aggregation attribute of type String that provides a way to indicate multiple values with a single expression. A compliant implementation MAY support the aggregation attribute if the Feature element is supported. Possible values for this attribute include:
  - List a comma-separated list of features
  - Pattern a regular expression (perl5) matching desired features
  - Range a range of nodes of the form: `<prefix>[5-23,77]`
- If an aggregation attribute is specified with the value of List, this element MAY also have a delimiter attribute of type String that indicates what delimiter is used to separate list elements. The default list delimiter is a comma.

- This element MAY have a count attribute of type Integer that indicates the instance count of the specified node(s).

The following is another example of a Node element:

```
<Node aggregation="Pattern">node[1-5]</Node>
```

## Appendix A

### Units of Measure Abbreviations

Abbreviation	Definition	Quantity
<b>B</b>	byte	1 byte
<b>KB</b>	Kilobyte	2 <sup>10</sup> bytes
<b>MB</b>	Megabyte	2 <sup>20</sup> bytes
<b>GB</b>	Gigabyte	2 <sup>30</sup> bytes
<b>TB</b>	Terabyte	2 <sup>40</sup> bytes
<b>PB</b>	Petabyte	2 <sup>50</sup> bytes
<b>EB</b>	Exabyte	2 <sup>60</sup> bytes
<b>ZB</b>	Aettabyte	2 <sup>70</sup> bytes
<b>YB</b>	Yottabyte	2 <sup>80</sup> bytes
<b>NB</b>	Nonabyte	2 <sup>90</sup> bytes
<b>DB</b>	Doggabyte	2 <sup>100</sup> bytes

## Scalable Systems Software Resource Management and Accounting Protocol (SSSRMAP) Wire Protocol

Resource Management Interface Specs  
 Release v. 3.0.3  
 13 May 2004

Scott Jackson  
 Brett Bode

## Status of This Memo

This is a specification defining a wire level protocol used between Scalable Systems Software components. It is intended that this specification will continue to evolve as these interfaces are implemented and thoroughly tested by time and experience.

## Abstract

This document is a specification describing a connection-oriented XML-based application layer client-server protocol for the interaction of resource management and accounting software components developed as part of the Scalable Systems Software Center. The SSSRMAP Wire Protocol defines a framing protocol that includes provisions for security. The protocol is specified in XML Schema Definition and rides on the HTTP protocol.

## Table of Contents

- [Scalable Systems Software Resource Management and Accounting Protocol \(SSSRMAP\) Wire Protocol](#)
- [Table of Contents](#)
- [1.0 Introduction](#)
- [2.0 Conventions Used in this Document](#)
  - [2.1 Keywords](#)
  - [2.2 XML Case Conventions](#)
  - [2.3 Schema Definitions](#)
- [3.0 Encoding](#)
  - [3.1 Schema Header and Namespaces](#)
  - [3.2 The Envelope Element](#)
  - [3.3 The Body Element](#)
- [4.0 Transport Layer](#)
- [5.0 Framing](#)
  - [5.1 Message Header Requirements](#)
  - [5.2 Message Chunk Format](#)
  - [5.3 Reply Header Requirements](#)
  - [5.4 Reply Chunk Format](#)

- [5.5 Message and Reply Tail Requirements and Multiple Chunks](#)
- [5.6 Examples](#)
  - [5.6.1 Sample SSSRMAP Message Embedded in HTTP Request](#)
  - [5.6.2 Sample SSSRMAP Reply Embedded in HTTP Response](#)
- [6.0 Asynchrony](#)
- [7.0 Security](#)
  - [7.1 Security Token](#)
    - [7.1.1 The SecurityToken Element](#)
    - [7.1.2 Security Token Types](#)
      - [7.1.2.1 Symmetric Key](#)
      - [7.1.2.2 Asymmetric Key](#)
      - [7.1.2.3 Password](#)
      - [7.1.2.4 Cleartext](#)
      - [7.1.2.5 Kerberos](#)
      - [7.1.2.6 GSI \(X.509\)](#)
    - [7.1.3 Example](#)
  - [7.2 Authentication](#)
    - [7.2.1 The Signature Element](#)
    - [7.2.2 The DigestValue Element](#)
    - [7.2.3 The SignatureValue Element](#)
    - [7.2.4 Signature Example](#)
  - [7.3 Confidentiality](#)
    - [7.3.1 The EncryptedData Element](#)
    - [7.3.2 The EncryptedKey Element](#)
    - [7.3.3 The CipherValue Element](#)
    - [7.3.4 Encryption Example](#)
- [8.0 Acknowledgements](#)
- [9.0 References](#)

## 1.0 Introduction

A major objective of the Scalable Systems Software [SSS] Center is to create a scalable and modular infrastructure for resource management and accounting on terascale clusters including resource

scheduling, grid-scheduling, node daemon support, comprehensive usage accounting and user interfaces emphasizing portability to terascale vendor operating systems. Existing resource management and accounting components feature disparate APIs (Application Programming Interfaces) requiring various forms of application coding to interact with other components.

This document proposes a wire level protocol expressed in an XML envelope to be considered as the foundation of a standard for communications between and among resource management and accounting software components. Individual components additionally need to define the particular XML binding necessary to represent the message format for communicating with the component.

## 2.0 Conventions Used in this Document

### 2.1 Keywords

The keywords **MUST**, **MUST NOT**, **REQUIRED**, **SHALL**, **SHALL NOT**, **SHOULD**, **RECOMMENDED**, **MAY**, and **OPTIONAL** in this document are to be interpreted as described in [RFC2119](#).

### 2.2 XML Case Conventions

In order to enforce a consistent capitalization and naming convention across all SSSRMAP specifications “Upper Camel Case” (UCC) and “Lower Camel Case” (LCC) Capitalization styles shall be used. UCC style capitalizes the first character of each word and compounds the name. LCC style capitalizes the first character of each word except the first word. [XML\_CONV][FED\_XML]

1. SSSRMAP XML Schema and XML instance documents **SHALL** use the following conventions:

- Element names **SHALL** be in UCC convention (example: <UpperCamelCaseElement/>).
- Attribute names **SHALL** be in LCC convention (example: <UpperCamelCaseElement lowerCamelCaseAttribute="Whatever"/>).

2. General rules for all names are:

- Acronyms **SHOULD** be avoided, but in cases where they are used, the capitalization **SHALL** remain (example: XMLSignature).
- Underscores (`_`), periods (`.`) and dashes (`-`) **MUST NOT** be used (example: use JobId instead of JOB.ID, Job\_ID or job-id).

### 2.3 Schema Definitions

SSSRMAP Schema Definitions appear like this

In case of disagreement between the schema file and this specification, the schema file takes precedence.

## 3.0 Encoding

Encoding tells how a message is represented when exchanged. SSSRMAP data exchange messages **SHALL** be defined in terms of XML schema [XML\_SCHEMA].

### 3.1 Schema Header and Namespaces

The header of the schema definition is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
  xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:sssrmmap="http://www.scidac.org/ScalableSystems/SSSRMAP"
  targetNamespace="http://www.scidac.org/ScalableSystems/SSSRMAP"
  elementFormDefault="qualified">
```

### 3.2 The Envelope Element

SSSRMAP messages and replies are encapsulated in the `Envelope` element. There are two possibilities for the contents of this element. If the contents are unencrypted, this element **MUST** contain a `Body` element and **MAY** contain a `Signature` element (refer to the section on [Security](#)). If the contents are encrypted, this element **MUST** contain exactly one `EncryptedData` element (refer to the section on [Security](#)). The `Envelope` element **MAY** contain namespace and other xsd-specific information necessary to validate the document against the schema. In addition, it **MAY** have any of the following attributes which may serve as processing clues to the parser:

Attribute	Description
<b>type</b>	A message type providing a hint as to the body contents such as “Request” or “Notification”
<b>component</b>	A component type such as “QueueManager” or “LocalScheduler”
<b>name</b>	A component name such as “OpenPBS” or “Maui”
<b>version</b>	A component version such as “2.2p12” or “3.2.2”

```
<complexType name=EnvelopeType">
  <choice minOccurs="1" maxOccurs="1">
    <choice minOccurs="1" maxOccurs="2">
      <element ref="sssrmmap:Signature" minOccurs="0" maxOccurs="1"/>
      <element ref="sssrmmap:Body" minOccurs="1" maxOccurs="1"/>
    </choice>
    <element ref="sssrmmap:EncryptedData" minOccurs="1" maxOccurs="1"/>
  </choice>
  <attribute name="type" type="string" use="optional"/>
  <attribute name="component" type="string" use="optional"/>
  <attribute name="name" type="string" use="optional"/>
  <attribute name="version" type="string" use="optional"/>
</complexType>

<element name="Envelope" type="sssrmmap:EnvelopeType"/>
```

### 3.3 The Body Element

- SSSRMAP messages and replies are encapsulated in the `Body` element. This element **MUST** contain exactly one `Request` or `Response` element.

```
<complexType name="BodyType">
  <choice minOccurs="1" maxOccurs="1">
    <element ref="sssrmap:Request" minOccurs="0" maxOccurs="1"/>
    <element ref="sssrmap:Response" minOccurs="0" maxOccurs="1"/>
    <any minOccurs="0" maxOccurs="1" namespace="##other"/>
  </choice>
</complexType>

<element name="Body" type="sssrmap:BodyType"/>
```

### 4.0 Transport Layer

This protocol will be built over the connection-oriented reliable transport layer TCP/IP. Support for other transport layers could also be considered, but native support for TCP/IP can be found on most terascale clusters and automatically handles issues such as reliability and connection fullness for the application developer implementing the SSSRMAP protocol.

### 5.0 Framing

Framing specifies how the beginning and ending of each message is delimited. Given that the encoding will be expressed as one or more XML documents, clients and servers need to know when an XML document has been fully read in order to be parsed and acted upon.

SSSRMAP uses the HTTP 1.1 [HTTP] protocol for framing. HTTP uses a byte-counting mechanism to delimit the message segments. HTTP chunked encoding is used. This allows for optional support for batched messages, large message segmentation and persistent connections.

#### 5.1 Message Header Requirements

The HTTP request line (first line of the HTTP request header) begins with POST and is followed by a URI and the version of the HTTP protocol that the client understands. It is suggested for this protocol that the URI consist of a single slash, followed by the protocol name in uppercase (i.e. /SSSRMAP), though this field is not checked and could be empty, a single slash or any URI.

The Content-Type must be specified as test/xml. Charset may be optionally specified and defaults to US-ASCII. It is recommended that charset be specified as "utf-8" for maximum interoperability.

The Transfer-Encoding must be specified as chunked. The Content-Length must NOT be specified as the chunk size is specified in the message chunk.

Other properties such as User-Agent, Host and Date are strictly optional.

#### 5.2 Message Chunk Format

A message chunk consists of a chunk size in hexadecimal format (whose value is the number of bytes in the XML message not including the chunk size and delimiter) delimited by a CR/LF "\r\n" and followed

by the message payload in XML that consists of a single XML document having a root element of `Envelope`.

### 5.3 Reply Header Requirements

The HTTP response line (first line of the HTTP response header) begins with HTTP and a version number, followed by a numeric code and a message indicating what sort of response is made. These response codes and messages indicate the status of the entire response and are as defined by the HTTP standard. The most common response is 200 OK, indicating that the message was received and an appropriate response is being returned.

The Content-Type must be specified as `text/xml`. Charset may be optionally specified and defaults to US-ASCII. It is recommended that charset be specified as “utf-8” for maximum interoperability.

The Transfer-Encoding MUST be specified as `chunked`. The Content-Length must NOT be specified.

Other properties such as Server, Host and Date are strictly optional.

### 5.4 Reply Chunk Format

A reply chunk consists of a chunk size in hexadecimal format (whose value is the number of bytes in the XML reply not including the chunk size and delimiter) delimited by a CR/LF “\r\n” and followed by the reply payload in XML that consists of a single XML document having a root element of `Envelope`.

### 5.5 Message and Reply Tail Requirements and Multiple Chunks

This specification only requires that single chunks be supported. A server may optionally be configured to handle requests with persistent connections (multiple chunks). It will be the responsibility of clients to know whether a particular server supports this additional functionality. After all chunks have been sent, a connection is terminated by sending a zero followed by a carriage return-linefeed combination (0\r\n) and closing the connection.

### 5.6 Examples

#### *Sample SSSRMAP Message Embedded in HTTP Request*

```
POST /SSSRMAP HTTP/1.1\r\n
Content-Type: text/xml; charset="utf-8"\r\n
Transfer-Encoding: chunked\r\n
\r\n
9A\r\n
<Envelope .../>
0\r\n
```

#### *Sample SSSRMAP Reply Embedded in HTTP Response*

```
HTTP/1.1 200 OK\r\n
Content-Type: text/xml; charset="utf-8"\r\n
Transfer-Encoding: chunked\r\n
\r\n
2B4\r\n
<Envelope .../>
0\r\n
```

## 6.0 Asynchrony

Asynchrony (or multiplexing) allows for the handling of independent exchanges over the same connection. A widely-implemented approach is to allow pipelining (or boxcarring) by aggregating requests or responses within the body of the message or via persistent connections and chunking in HTTP 1.1. Pipelining helps reduce network latency by allowing a client to make multiple requests of a server, but requires the requests to be processed serially [RFC3117]. Parallelism could be employed to further reduce server latency by allowing multiple requests to be processed in parallel by multi-threaded applications.

Segmentation may become necessary if the messages are larger than the available window. With support for segmentation, the octet-counting requirement that you need to know the length of the whole message before sending it can be relegated to the segment level – and you can start sending segments before the whole message is available. Segmentation is facilitated via “chunking” in HTTP 1.1.

The current SSSRMAP strategy supports only a single request or response within the Body element. A server may optionally support persistent connections from a client via HTTP chunking. Segmentation of large responses is also optionally supported via HTTP chunking. Later versions of the protocol could allow pipelined requests and responses in a single Body element.

## 7.0 Security

SSSRMAP security features include capabilities for integrity, authentication, confidentiality, and non-repudiation. The absence or presence of the various security features depend upon the type of security token used and the protection methods you choose to specify in the request.

For compatibility reasons, SSSRMAP specifies six supported security token types. Extensibility features are included allowing an implementation to use alternate security algorithms and security tokens. It is also possible for an implementation to ignore security features if it is deemed nonessential for the component. However, it is highly RECOMMENDED that an implementation support at least the default security token type in both authentication and encryption.

### 7.1 Security Token

A security token may be included in either the Signature block, and/or in the EncryptedData block (both described later) as an implicit or explicit cryptographic key. If this element is omitted, the security token is assumed to be a secret key shared between the client and the server.

#### *The SecurityToken Element*

This element is of type String. If the security token conveys an explicit key, this element’s content is the value of the key. If the key is natively expressed in a binary form, it must be converted to base64 encoding as defined in XML Digital Signatures (“<http://www.w3.org/2000/09/xmldsig#base64>”). If the type is not specified, it is assumed to be of type “Symmetric”.

It may have any of the following optional attributes:

Attribute	Description
<b>type</b>	<p>The type of security token (described subsequently)</p> <ul style="list-style-type: none"> <li>• A <code>type</code> attribute of “Symmetric” specifies a shared secret key between the client and server. This is the default.</li> <li>• A <code>type</code> attribute of “Asymmetric” specifies the use of public private key pairs between the client and server.</li> <li>• A <code>type</code> attribute of “Password” encrypts and authenticates with a user password known to both the client and server.</li> <li>• A <code>type</code> attribute of “Cleartext” allows the passing of a cleartext username and password and depends on the use of a secure transport (such as SSL or IPsec).</li> <li>• A <code>type</code> attribute of “Kerberos5” specifies a kerberos token.</li> <li>• A <code>type</code> attribute of “X509v3” specifies an X.509 certificate.</li> </ul>
<b>name</b>	<p>The name of the security token which serves as an identifier for the actor making the request (useful when the key is a password, or when the key value is implicit as when a public key is named but not included)</p>

```

<complexType name="SecurityTokenType" mixed="true">
  <simpleContent>
    <extension base="string">
      <attribute name="type" type="string" use="optional">
        <attribute name="name" type="string" use="optional">
        </extension>
      </simpleContent>
    </complexType>
  </complexType>

<element name="SecurityToken" type="sssrmap:SecurityTokenType"/>

```

## Security Token Types

SSSRMAP defines six standard security token types:

### Symmetric Key

The default security token specifies the use of a shared secret key. The secret key is up to 128-bits long and known by both client and server. When using a symmetric key as a security token, it is not necessary to specify the `type` attribute with value “Symmetric” because this is assumed when the attribute is absent. The `name` attribute should be specified indicating the actor issuing the request. If the user provides a password to be sent to the server for authentication, then the password is encrypted with the secret key using a default `method="kw-tripledes"` (XML ENCRYPTION <http://www.w3.org/2001/04/xmlenc#kw-tripledes>), base64 encoded and included as the string content of the `SecurityToken` element. If the client authenticated the user, then the `SecurityToken` element is empty. The same symmetric key is used in both authentication and encryption.

### Asymmetric Key

Public and private key pairs can be used to provide non-repudiation of the client (or server). The client and the server must each have their own asymmetric key pairs. This mode is indicated by specifying the `type` attribute as “Asymmetric”. The `name` attribute should be specified indicating the actor issuing the request. If the user provides a password to be sent to the server for authentication, then the password is

encrypted with the server's public key using a default `method="rsa-1_5"` (XML ENCRYPTION [http://www.w3.org/2001/04/xmlenc#rsa-1\\_5](http://www.w3.org/2001/04/xmlenc#rsa-1_5)), base64 encoded and included as the string content of the `SecurityToken` element. If the client authenticated the user, then the `SecurityToken` element is empty. The sender's private key is used in authentication (signing) while the recipient's public key is used for encryption.

### Password

This mode allows for a username password combination to be used under the assumption that the server also knows the password for the user. This security token type is indicated by specifying a value of "Password" for the `type` attribute. The password itself is used as the cryptographic key for authentication and encryption. The `name` attribute contains the user name of the actor making the request. The `SecurityToken` element itself is empty.

### Cleartext

This security mode is equivalent to passing the username and password in the clear and depends upon the use of a secure transport (such as SSL or IPsec). The purpose of including this security token type is to enable authentication to occur from web browsers over SSL or over internal LANs who use IPsec to encrypt all traffic. The password (or a hash of the password like in `/etc/passwd`) would have to be known by the server for authentication to occur. In this mode, neither encryption nor signing of the hash is performed at the application layer. This mode is indicated by specifying a value of "Cleartext" for the `type` attribute. The `name` attribute contains the user name of the actor making the request and the string content of the `SecurityToken` element is the unencrypted plaintext password.

### Kerberos

The use of a Kerberos version 5 token is indicated by specifying "Kerberos5" in the `type` attribute. The `name` attribute is used to contain the kerberos user id of the actor making the request. The `SecurityToken` element contains two sub elements. The `Authenticator` element contains the authenticator encoded in base64. A `Ticket` element contains the service-granting ticket, also base64 encoded.

### GSI (X.509)

The Grid Security Infrastructure (GSI) which is based on public key encryption, X.509 certificates, and the Secure Sockets Layer (SSL) communication protocol can be indicated by specifying a `type` attribute of "X509v3". The `name` attribute contains the userid used that the actor was mapped to in the local system. The string content of the `SecurityToken` element is the GSI authentication message including the X.509 identity of the sender encoded in base64.

### Example

```
<SecurityToken type="Asymmetric" name="scottmo">
MIIEZzCCA9CggAwIBAgIQEmtJZc0rqrKh5i...
</SecurityToken>
```

## 7.2 Authentication

Authentication entails how the peers at each end of the connection are identified and verified. Authentication is optional in an SSSRMAP message or reply. SSSRMAP uses a digital signature scheme for authentication that borrows from concepts in XML Digital Signatures [XML\_DSIG]. In addition to

authentication, the use of digital signatures also ensures integrity of the message, protecting exchanges from third-party modification.

When authentication is used, a `Signature` element is prepended as the first element within the `Envelope` element. All of the security modes will create a digest of the data for integrity checking and store this in base64 encoding in a `DigestValue` element as a child of the `Signature` element. The digital signature is created by encrypting the hash with the appropriate security token and storing this value in a `SignatureValue` element as a child of the `Signature` element. The security token itself is included as a child of the `Security` element within a `SecurityToken` element.

There are a number of procedural practices that must be followed in order to standardize this approach. The digest (or hash) is created over the contents of the `Envelope` element (not including the `Element` tag or its attributes). This might be over one or more `Request` or `Notify` elements (or `Response` or `Ack` elements) and necessarily excludes the `Signature` Element itself. (Note that any data encryption is performed after the creation of the digital signature and any decryption is performed before authenticating so the `EncryptedData` element will not interfere with this process. Hence, the signature is always based on the (hashed but) unencrypted data). For the purposes of generating the digest over the same value, it is assumed that the data is first canonicalized to remove extraneous whitespace, comments, etc according to the XML Digital Signature algorithm ("<http://www.w3.org/TR/2001/REC-xml-c14n-20010315>") and a transform is applied to remove namespace information. As a rule, any binary values are always transformed into their base64 encoded values when represented in XML.

### The Signature Element

The `Signature` element **MUST** contain a `DigestValue` element that is used for integrity checking. It **MUST** also contain a `SecurityToken` element that is used to indicate the security mode and token type, and to verify the signature. It **MUST** contain a `SignatureValue` element that contains the base64 encrypted value of the signature wrought on the hash **UNLESS** the security token type indicates `Cleartext` mode where a signature would be of no value with the encryption key being sent in the clear -- in this case we use the password itself for authentication).

```
<complexType name="SignatureType">
  <choice minOccurs="2" maxOccurs="3">
    <element ref="sssrmmap:DigestValue" minOccurs="1" maxOccurs="1"/>
    <element ref="sssrmmap:SignatureValue" minOccurs="1" maxOccurs="1"/>
    <element ref="sssrmmap:SecurityToken" minOccurs="0" maxOccurs="1"/>
  </choice>
</complexType>

<element name="Signature" type="sssrmmap:SignatureType"/>
```

### The DigestValue Element

The `DigestValue` element contains the cryptographic digest of the message data. As described above, the hash is generated over the `Body` element. The data to be hashed must first be canonicalized and appropriately transformed before generating the digest since typically an application will read in the XML document into an internal binary form, then marshal (or serialize) the data into a string which is passed as input to the hash algorithm. Different implementations marshal the data differently so it is necessary to convert this to a well-defined format before generating the digest or the clients will generate different digest values for the same XML. The SHA-1 [SHA-1] message digest algorithm (<http://www.w3.org/2000/09/xmldsig#sha1>) SHALL be used as the default method for generating the

digest. A *method* attribute is defined as an extensibility option in case an implementation wants to be able to specify alternate message digest algorithms.

It MAY have a method attribute:

Attribute	Description
<b>method</b>	The message digest algorithm. <ul style="list-style-type: none"> <li>A <i>method</i> attribute of “sha1” specifies the SHA-1 message digest algorithm. This is the default and is implied if this attribute is omitted.</li> </ul>

```
<complexType name="DigestValueType">
  <simpleContent>
    <extension base="string">
      <attribute name="method" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="DigestValue" type="sssrmap:DigestValueType"/>
```

### The SignatureValue Element

The SignatureValue element contains the digital signature that serves the authentication (and potentially non-repudiation) function. The string content of the SignatureValue element is a base64 encoding of the encrypted digest value. The HMAC algorithm [HMAC] based on the SHA1 message digest (<http://www.w3.org/2000/09/xmldsig#hmac-sha1>) SHALL be used as the default message authentication code algorithm for user identification and message integrity. A *method* attribute is defined as an extensibility option in case an implementation wants to be able to specify alternate digital signature algorithms.

It MAY have a method attribute:

Attribute	Description
<b>method</b>	The digest signature algorithm. <ul style="list-style-type: none"> <li>A <i>method</i> attribute of “hmac-sha1” specifies the HMAC SHA-1 digital signature algorithm. This is the default and is implied if this attribute is omitted.</li> </ul>

```
<complexType name="SignatureValueType">
  <simpleContent>
    <extension base="string">
      <attribute name="method" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="SignatureValue" type="sssrmap:SignatureValueType"/>
```

### Signature Example

Pre-authentication:

```

<Envelope>
  <Body>
    <Request action="Query" actor="kenneth">
      <Object>User</Object>
      <Get name="EmailAddress"></Get>
      <Where name="Name">scott</Where>
    </Request>
  </Body>
</Envelope>

```

#### Post-authentication:

```

<Envelope>
  <Signature>
    <DigestValue>
      LyLsF0Pi4wPU...
    </DigestValue>
    <SignatureValue>
      DJbchm5gK...
    </SignatureValue>
    <SecurityToken type="Asymmetric" name="kenneth">
      MIIEZzCCA9CggAwIBAgIQEmtJZc0rqrKh5i...
    </SecurityToken>
  </Signature>
  <Body>
    <Request action="Query" actor="kenneth">
      <Object>User</Object>
      <Get name="EmailAddress"></Get>
      <Where name="Name">scottmo</Where>
    </Request>
  </Body>
</Envelope>

```

## 7.3 Confidentiality

Confidentiality involves encrypting the sensitive data in the message, protecting exchanges against third-party interception and modification. Confidentiality is optional in an SSSRMAP message or reply. When confidentiality is required, SSSRMAP sessions use block cipher encryption with concepts borrowed from the emerging XML Encryption [XML\_ENC] standard.

When confidentiality is used, encryption is performed over all child elements of the `Envelope` element, i.e. on the message data as well as any signature (The encrypted data is not signed -- rather the signature is encrypted). This data is replaced in-place within the envelope with an `EncryptedData` element. The data is first compressed using the gzip algorithm [ZIP]. Instead of encrypting this compressed data with the security token directly, a 192-bit random session key is generated by the sender and used to perform symmetric encryption on the compressed data. This key is itself encrypted with the security token and included with the encrypted data as the value of the `EncryptedKey` element as a child of the `EncryptedData` element. The ciphertext resulting from the data being encrypted with the session key is passed as the value of a `CipherValue` element (also a child of the `EncryptedData` element). As in the case with authentication, the security token itself is included as a child of the `Security` element within a `SecurityToken` element.

### The EncryptedData Element

When SSSRMAP confidentiality is required, the EncryptedData element MUST appear as the only child element in the Envelope element. It directly replaces the contents of these elements including the data and any digital signature. It MUST contain an EncryptedKey element that is used to encrypt the data. It MUST contain a CipherValue element that holds the base64 encoded ciphertext. It MAY also contain a SecurityToken element that is used to indicate the security mode and token type. If the SecurityToken element is omitted, a Symmetric key token type is assumed. Confidentiality is not used when a security token type of "Cleartext" is specified since it would be pointless to encrypt the data with the encryption key in the clear.

```
<complexType name="EncryptionDataType">
  <choice minOccurs="0" maxOccurs="1">
    <element ref="sssmap:EncryptedKey" minOccurs="1" maxOccurs="1"/>
    <element ref="sssmap:CipherValue" minOccurs="1" maxOccurs="1"/>
    <element ref="sssmap:SecurityToken" minOccurs="1" maxOccurs="1"/>
  </choice>
</complexType>

<element name="EncryptedData" type="sssmap:EncryptionDataType"/>
```

### The EncryptedKey Element

The EncryptedKey element is a random session key encrypted with the security token. This approach is used for a couple of reasons. In the case where public key encryption is used, asymmetric encryption is much slower than symmetric encryption and it makes sense to use a symmetric key for encryption and pass along it along by encrypting it with the recipient’s public key. It is also useful in that the security token which does not change very often (compared to the session key which changes for every connection) is used on a very small sampling of data (the session key), whereas if it was used to encrypt the whole message an attacker could more effectively exploit an attack against the ciphertext. The CMS Triple DES Key Wrap algorithm “kw-tripledes” SHALL be used as the default method for key encryption. The session key is encrypted using the security token, base64 encoded and specified as the string content of the EncryptedKey element. A method attribute is defined as an extensibility option in case an implementation wants to be able to specify alternate key encryption algorithms.

It is REQUIRED that an implementation use a cryptographically secure Pseudo-Random number generator. It is RECOMMENDED that the session key be cryptographically generated (such as cyclic encryption, DES OFB, ANSI X9.17 PRNG, SHA1PRNG, or ANSI X12.17 (used by PGP)).

It MAY have a method attribute:

Attribute	Description
<b>method</b>	<p>The key encryption algorithm.</p> <ul style="list-style-type: none"> <li>A method attribute of “kw-tripledes” specifies the CMS Triple DES Key Wrap algorithm. This algorithm is specified by the XML Encryption [XML_ENC] URI “http://www.w3.org/2001/04/xmlenc#kw-tripledes”. It involves two Triple DES encryptions, a random and known Initialization Vector (IV) and a CMS key checksum. A 192-bit key encryption key is generated from the security token, lengthened as necessary by zero-padding. No additional padding is performed in the encryptions. This is the default and is implied if this attribute is omitted.</li> </ul>

```

<complexType name="EncryptedKeyType">
  <simpleContent>
    <extension base="string">
      <attribute name="method" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="EncryptedKey" type="sssrmmap:EncryptedKeyType"/>

```

### The CipherValue Element

The CipherValue element contains the message (and possibly signature) data encrypted with the random session key. The ciphertext is compressed using the gzip algorithm [ZIP], encrypted by the designated method, base64 encoded and included as the string content of the CipherValue element. The Triple DES algorithm with Cipher Block Chaining (CBC) feedback mode SHALL be used as the default method for encryption. A *method* attribute is defined as an extensibility option in case an implementation wants to be able to specify alternate data encryption algorithms.

It MAY have a method attribute:

Attribute	Description
<b>method</b>	<p>The data encryption algorithm.</p> <ul style="list-style-type: none"> <li>A <i>method</i> attribute of "tripleDES-cbc" specifies the Triple DES algorithm with Cipher Block Chaining (CBC) feedback mode. This algorithm is specified by the XML Encryption [XML_ENC] URI identifier "<a href="http://www.w3.org/2001/04/xmlenc#tripleDES-cbc">http://www.w3.org/2001/04/xmlenc#tripleDES-cbc</a>". It specifies the use of a 192-bit encryption key and a 64-bit Initialization Vector (IV). Of the key bits, the first 64 are used in the first DES operation, the second 64 bits in the middle DES operation, and the third 64 bits in the last DES operation. The plaintext is first padded to a multiple of the block size (8 octets) using the padding scheme described in [XML_ENC] for Block Encryption Algorithms (Padding per PKCS #5 will suffice for this). The resulting cipher text is prefixed by the IV. This is the default and is implied if this attribute is omitted.</li> </ul>

```

<complexType name="CipherValueType">
  <simpleContent>
    <extension base="string">
      <attribute name="method" type="string" use="optional"/>
    </extension>
  </simpleContent>
</complexType>

<element name="CipherValue" type="sssrmmap:CipherValueType"/>

```

### Encryption Example

In this example, a simple request is demonstrated without a digital signature for the sake of emphasizing the encryption plaintext replacement.

Pre-encryption:

```

<Envelope>
  <Body>
    <Response>
      <Status>true</Status>
      <Code>000</Code>
      <Count>1</Count>
      <Data>
        <User>
          <EmailAddress>Scott.Jackson@pnl.gov</EmailAddress>
        </User>
      </Data>
    </Response>
  </Body>
</Envelope>

```

#### Post-encryption:

```

<Envelope>
  <EncryptedData>
    <EncryptedKey>
      NAKe9iQofYhyOfiHZ29kkEFVJ30CAwEAAaMSM...
    </EncryptedKey>
    <CipherValue>
      mPCadVfOMx1NzDaKMHNgFkR9upTW4kgBxyPW...
    </CipherValue>
    <SecurityToken type="Asymmetric" name="kenneth">
      MIIEZzCCA9CggAwIBAgIQEmtJZc0rqrKh5i...
    </SecurityToken>
  </EncryptedData>
</Envelope>

```

## 8.0 Acknowledgements

### 9.0 References

- [RFC2119] S. Bradner, "Key Words for Use in RFCs to Indicate Requirement Levels", [RFC 2119](#), March 1997.
- [BEEP] M. Rose, "The Blocks Extensible Exchange Protocol Core", [RFC 3080](#), March 2001.
- [HMAC] H. Krawczyk, M. Bellare, R. Canetti, "HMAC, Keyed-Hashing for Message Authentication", [RFC 2104](#), February 1997.
- [SHA-1] U.S. Department of Commerce/National Institute of Standards and Technology, "[Secure Hash Standard](#)", FIPS PUB 180-1.
- [SSS] "Scalable Systems Software", <http://www.scidac.org/ScalableSystems>
- [HTTP] "Hypertext Transfer Protocol – HTTP/1.1", [RFC 2616](#), June 1999.
- [XML\_CONV] "[I-X and <I-N-CA> XML Conventions](#)".
- [FED\_XML] "[U.S. Federal XML Guidelines](#)".
- [RFC3117] M. Rose, "On the Design of Application Protocols", [Informational RFC 3117](#), November 2001.
- [XML\_DSIG] D. Eastlake, J. Reagle Jr., D. Solo, "[XML Signature Syntax and Processing](#)", W3C Recommendation, 12 February 2002.

[XML\_ENC] T. Imamura, B. Dillaway, E. Smon, "[XML Encryption Syntax and Processing](#)", W3C Candidate Recommendation, 4 March 2002.

[XRP] E. Brunner-Williams, A. Damaraju, N. Zhang, "[Extensible Registry Protocol \(XRP\)](#)", Internet Draft, expired August 2001.

[XML] Bray, T., et al, "[Extensible Markup Language \(XML\) 1.0 \(Second Edition\)](#)", 6 October 2000.

[XML\_SCHEMA] D. Beech, M. Maloney, N. Mendelshohn, "[XML Schema Part 1: Structures Working Draft](#)", April 2000.

[ZIP] J. Gailly, M. Adler, "The gzip home page", <http://www.gzip.org/>

## Appendix W: Moab Resource Manager Language Interface Overview

The Moab RM Language (formerly called WIKI) is the language that some resource managers use to communicate with Moab, specifically a native RM. Generally each line represents a single resource or workload in Moab. The line contains the name of the resource or workload followed by a set of `<attr>=<val>` pairs. Although the Moab RM language follows the same data format for all RMs, each RM type receives and returns it differently. For instructions and examples on using Moab RM language with SLURM or a native RM, see [W.2 Managing Resources with SLURM on page 1372](#) and [Managing Resources Directly with the Native Interface on page 650](#) respectively.

- [W.1 Moab Resource Manager Language Data Format](#)
- [W.2 Managing Resources with SLURM](#)
- [W.3 Moab RM Language Socket Protocol Description](#)

### W.1 Moab Resource Manager Language Data Format

- [W.1.1 Query Resources Data Format](#)
- [W.1.2 Query Workload Data Format](#)

#### W.1.1 Query Resources Data Format

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>ADISK</b>	<code>&lt;INTEGER&gt;</code>	0	Available local disk on node (in MB)
<b>AFS</b>	<code>&lt;fs id="X" size="X" io="Y" rcount="X" wcount="X" ocoun- t="X"&gt;&lt;/fs&gt; [...]</code>	0	Available filesystem state

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>AMEMORY</b>	<INTEGER>	0	Available/free RAM on node (in MB)
<b>APROC</b>	<INTEGER>	1	Available processors on node
<b>ARCH</b>	<STRING>	---	Compute architecture of node
<b>ARES</b>	one or more comma delimited <NAME>: <VALUE> pairs (ie, MATLAB: 6, COMPILER: 100)	---	Arbitrary consumable resources currently available on the node
<b>ASWAP</b>	<INTEGER>	0	Available swap on node (in MB)
<b>CCLASS</b>	one or more bracket enclosed <NAME>: <COUNT> pairs (ie, [batch:5] [sge:3])	---	Run classes supported by node. Typically, one class is 'consumed' per task. Thus, an 8 processor node may have 8 instances of each class it supports present, ie [batch:8] [interactive:8]
<b>CDISK</b>	<INTEGER>	0	Configured local disk on node (in MB)
<b>CFS</b>	<STRING>	0	Configured filesystem state
<b>CMEMORY</b>	<INTEGER>	0	Configured RAM on node (in MB)
<b>CONTAINERNODE</b>	<STRING>	---	The physical machine that is host- ing the virtual machine. Only valid on VMs.
<b>CPROC</b>	<INTEGER>	1	Configured processors on node
<b>CPULOAD</b>	<DOUBLE>	0.0	One minute BSD load average
<b>CPUSPEED</b>	<INTEGER>	---	The node's processor speed in MHz

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>CRES</b>	one or more comma delimited <NAME>: <VALUE> pairs (ie, MATLAB:6, COMPILER:100)	---	Arbitrary consumable resources supported and tracked on the node, ie software licenses or tape drives
<b>CSWAP</b>	<INTEGER>	0	Configured swap on node (in MB)
<b>FEATURE</b>	one or more colon delimited <STRING>'s (ie, WIDE:HSM)	---	Generic attributes, often describing hardware or software features, associated with the node
<b>GEVENT</b>	GEVENT [ <EVENTNAME> ] = <STRING>	---	<a href="#">Generic event</a> occurrence and context data
<b>GMETRIC</b>	GMETRIC [ <METRICNAME> ] = <DOUBLE>	---	Current value of <a href="#">generic metric</a> , i.e., 'GMETRIC [temp]=103.5'.
<b>IDLETIME</b>	<INTEGER>	---	Number of seconds since last detected keyboard or mouse activity (often used with <a href="#">desktop harvesting</a> )
<b>MAXTASK</b>	<INTEGER>	<CPROC>	Maximum number of tasks allowed on the node at any given time
<b>NETADDR</b>	<STRING>	---	The IP address of the machine
<b>NODEINDEX</b>	<INTEGER>	---	The <a href="#">node's index</a>
<b>OS</b>	<STRING>	---	Operating system running on node
<b>OSLIST</b>	One or more comma delimited <STRING>'s with quotes if the string has spaces (ie. "SAS7 AS3 Core Baseline Build v0.1.0", "RedHat AS3-U5Development Build v0.2").	---	Operating systems accepted by node

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>OTHER</b>	<ATTR>=<VALUE> [, <ATTR>=<VALUE>] . . .	---	Opaque node attributes assigned to node
<b>PARTITION</b>	<STRING>	DEFAULT	Partition to which node belongs
<b>POWER</b>	<BOOLEAN>		Whether the machine is on or off
<b>PRIORITY</b>	<INTEGER>	---	<a href="#">Node allocation priority</a>
<b>RACK</b>	<INTEGER>	0	Rack location of the node
<b>SLOT</b>	<INTEGER>	0	Slot location of the node
<b>STATE*</b>	one of the following: Idle, Running, Busy, Unknown, Drained, Draining, or Down	Down	State of the node
<b>UPDATETIME*</b>	<EPOCHTIME>	0	Time node information was last updated
<b>VARATTR</b>	<ATTR1>=<VAL1>[=<display- name1>] [+<ATTR2>=<VAL2> [=<display- name2>]] . . .	---	<p>Plus-delimited (+) list of &lt;ATTR&gt;=&lt;VAL&gt; [=&lt;displayName&gt;] pairs that jobs can request. You can replace any of the equals signs with colons if desired.</p> <p>Specifying a display name allows you to choose a name that will be displayed in the Mongo database instead of the unique ID (the &lt;VALUE&gt;).</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> If you give two different attributes the same value and one of them also has a display name specified, both attributes will appear with the same display name.</p> </div>
<b>VARIABLE</b>	<ATTR>=<VAL>	---	Generic variables to be associated with node

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>VMOSLIST</b>	<STRING>	---	Comma-delimited list (,) of supported virtual machine operating systems for this node
<b>XRES</b>	one or more comma delimited <NAME>: <VALUE> pairs (ie, MATLAB:6, COMPILER:100)	---	Amount of external usage of a particular generic resource

\* indicates required field

Node states have the following definitions:

State	Description
<b>Busy</b>	Node is running some jobs and will not accept additional jobs
<b>Down</b>	Resource Manager problems have been detected. Node is incapable of running jobs.
<b>Draining</b>	Node is responding but will not accept new jobs
<b>Idle</b>	Node is ready to run jobs but currently is not running any.
<b>Running</b>	Node is running some jobs and will accept additional jobs
<b>Unknown</b>	Node is capable of running jobs but the scheduler will need to determine if the node state is actually Idle, Running, or Busy.

## W.1.2 Query Workload Data Format

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>ACCOUNT</b>	<STRING>	---	AccountID associated with job
<b>ARGS</b>	<STRING>	---	job command-line arguments
<b>COMMENT</b>	<STRING>	0	job <a href="#">resource manager extension</a> arguments including qos, dependencies, reservation constraints, etc
<b>COMPLETETIME*</b>	<EPOCHTIME>	0	time job completed execution

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>DDISK</b>	<INTEGER>	0	quantity of local disk space (in MB) which must be dedicated to each task of the job
<b>DGRES</b>	name:value [,name:value]	---	Dedicated generic resources per task.
<b>DPROCS</b>	<INTEGER>	1	number of processors dedicated per task
<b>DSWAP</b>	<INTEGER>	0	quantity of virtual memory (swap, in MB) which must be dedicated to each task of the job
<b>ENDDATE</b>	<EPOCHTIME>	[ANY]	time by which job must complete
<b>ENV</b>	<STRING>	---	job environment variables
<b>ERROR</b>	<STRING>	---	file to contain STDERR
<b>EVENT</b>	<EVENT>	---	event or exception experienced by job
<b>EXEC</b>	<STRING>	---	job executable command
<b>EXITCODE</b>	<INTEGER>	---	job exit code
<b>FLAGS</b>	<STRING>	---	job flags
<b>GEOMETRY</b>	<STRING>	---	String describing task geometry required by job
<b>GNAME*</b>	<STRING>	---	GroupID under which job will run

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>HOSTLIST</b>	comma or colon delimited list of hostnames - suffix the hostlist with a caret (^) to mean superset; suffix with an asterisk (*) to mean subset; otherwise, the hostlist is interpreted as an exact set	[ANY]	list of required hosts on which job must run. (see <a href="#">TASKLIST</a> ) A subset means the specified hostlist is used first to select hosts for the job. If the job requires more hosts than are in the hostlist, they will be obtained from elsewhere if possible. If the job does not require all of the jobs in the hostlist, it will use only the ones it needs. A superset means the hostlist is the <i>only</i> source of hosts that should be considered for running the job. If the job can't find the necessary resources in the hosts in this list it should <i>not</i> run. No other hosts should be considered in allocating the job.
<b>INPUT</b>	<STRING>	---	file containing STDIN
<b>IWD</b>	<STRING>	---	job's initial working directory
<b>NAME</b>	<STRING>	---	User specified name of job
<b>NODES</b>	<INTEGER>	1	Number of nodes required by job (See <a href="#">Node Definition</a> for more info)
<b>OUTPUT</b>	<STRING>	---	file to contain STDOUT
<b>PARTITIONMASK</b>	one or more colon delimited <STRING>s	[ANY]	list of partitions in which job can run
<b>PREF</b>	colon delimited list of <STRING>s	---	List of preferred node features or variables. (See <a href="#">PREF</a> for more information.)
<b>PRIORITY</b>	<INTEGER>	---	system priority (absolute or relative - use '+' and '-' to specify relative)
<b>QOS</b>	<INTEGER>	0	quality of service requested
<b>QUEUETIME*</b>	<EPOCHTIME>	0	time job was submitted to resource manager
<b>RARCH</b>	<STRING>	---	architecture required by job

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>RCLASS</b>	list of bracket enclosed <STRING> :<INTEGER> pairs	---	list of <CLASSNAME>:<COUNT> pairs indicating type and number of class instances required per task. (ie, [batch:1] or [batch:2] [tape:1])
<b>RDISK</b>	<INTEGER>	0	local disk space (in MB) required to be configured on nodes allocated to the job
<b>RDISKCMP</b>	one of >=, >, ==, <, or <=	>=	local disk comparison (ie, node must have > 2048 MB local disk)
<b>REJCODE</b>	<INTEGER>	0	reason job was rejected
<b>REJCOUNT</b>	<INTEGER>	0	number of times job was rejected
<b>REJMESSAGE</b>	<STRING>	---	text description of reason job was rejected
<b>REQRSV</b>	<STRING>	---	Name of reservation in which job must run
<b>RESACCESS</b>	<STRING>	---	List of reservations in which job can run
<b>RFEATURES</b>	colon delimited list <STRING>'s	---	List of features required on nodes
<b>RMEM</b>	<INTEGER>	0	real memory (RAM, in MB) required to be configured on nodes allocated to the job
<b>RMEMCMP</b>	one of '>=', '>', '==', '<', or '<='	>=	real memory comparison (ie, node must have >= 512MB RAM)
<b>ROPSYS</b>	<STRING>	---	operating system required by job
<b>RSOFTWARE</b>	<RESTYPE>[{:+}] <COUNT>] [@<TIMEFRAME>]	---	software required by job
<b>RSWAP</b>	<INTEGER>	0	virtual memory (swap, in MB) required to be configured on nodes allocated to the job

NAME	FORMAT	DEFAULT	DESCRIPTION
<b>RSWAPCMP</b>	one of '>=', '>', '==', '<', or '<='	>=	virtual memory comparison (ie, node must have ==4096 MB virtual memory)
<b>SID</b>	<STRING>	---	system id (global job system owner)
<b>STARTDATE</b>	<EPOCHTIME>	0	earliest time job should be allowed to start
<b>STARTTIME*</b>	<EPOCHTIME>	0	time job was started by the resource manager
<b>STATE*</b>	one of <a href="#">Idle</a> , Running, Hold, Suspended, Completed, or Removed	Idle	State of job
<b>SUSPENDTIME</b>	<INTEGER>	0	Number of seconds job has been suspended
<b>TASKLIST</b>	one or more comma-delimited <STRING>'s	---	list of <b>allocated</b> tasks, or in other words, comma-delimited list of node ID's associated with each active task of job (i.e., cl01, cl02, cl01, cl02, cl03) The tasklist is initially selected by the scheduler at the time the StartJob command is issued. The resource manager is then responsible for starting the job on these nodes and maintaining this task distribution information throughout the life of the job. (see <a href="#">HOSTLIST</a> )
<b>TASKS*</b>	<INTEGER>	1	Number of tasks required by job (See <a href="#">Task Definition</a> for more info)
<b>TASKPERNODE</b>	<INTEGER>	0	exact number of tasks required per node
<b>UNAME*</b>	<STRING>	---	UserID under which job will run
<b>UPDATETIME*</b>	<EPOCHTIME>	0	Time job was last updated
<b>WCLIMIT*</b>	[[HH:]MM:]SS	864000	walltime required by job

\* indicates required field

Job states have the following definitions:

State	Definition
<b>Completed</b>	Job has completed
<b>Hold</b>	Job is in the queue but is not allowed to run
<b>Idle</b>	Job is ready to run
<b>Removed</b>	Job has been canceled or otherwise terminated externally
<b>Running</b>	Job is currently executing
<b>Suspended</b>	job has started but execution has temporarily been suspended

**i** Completed and canceled jobs should be maintained by the resource manager for a brief time, perhaps 1 to 5 minutes, before being purged. This provides the scheduler time to obtain all final job state information for scheduler statistics.

#### Related topics

- [Managing Resources with SLURM](#)
- [Managing Resources Directly with the Native Interface](#)

## W.2 Managing Resources with SLURM

This section demonstrates how Moab uses the Moab RM language (formerly called WIKI) to communicate with SLURM. For SLURM configuration instructions, see the [Moab-SLURM Integration Guide](#).

- [W.2.1 Commands](#)
  - [W.2.1.1 Resource Query](#)
    - [W.2.1.1.1 Query Resources Request Format](#)
    - [W.2.1.1.2 Query Resources Response Format](#)
  - [W.2.1.2 Workload Query](#)
    - [W.2.1.2.1 Query Workload Request Format](#)
    - [W.2.1.2.2 Query Workload Response Format](#)
    - [W.2.1.2.3 Query Workload Example](#)
  - [W.2.1.3 Start Job](#)
  - [W.2.1.4 Cancel Job](#)
  - [W.2.1.5 Suspend Job](#)

- [W.2.1.6 Resume Job](#)
- [W.2.1.7 Requeue Job](#)
- [W.2.1.8 Signal Job](#)
- [W.2.1.9 Modify Job](#)
- [W.2.1.10 JobAddTask](#)
- [W.2.1.11 JobRemoveTask](#)
- [W.2.2 Rejection Codes](#)

## W.2.1 Commands

All commands are requested via a socket interface, one command per socket connection. All fields and values are specified in ASCII text.

Supported Commands are:

- [Query Resources](#)
- [Query Workload](#)
- [Start Job](#)
- [Cancel Job](#)
- [Suspend Job](#)
- [Resume Job](#)
- [Requeue Job](#)
- JOBADDTASK
- JOBRELEASETASK

### W.2.1.1 Moab RM Language Query Resources

#### W.2.1.1.1 Moab RM Language Query Resources Request Format

```
CMD=GETNODES ARG={ <UPDATETIME>:<NODEID>[:<NODEID>] . . . | <UPDATETIME>:ALL }
```

Only nodes updated more recently than *<UPDATETIME>* will be returned where *<UPDATETIME>* is specified as the epoch time of interest. Setting *<UPDATETIME>* to 0 will return information for all nodes. Specify a colon delimited list of *NODEIDs* if specific nodes are desired or use the keyword *ALL* to receive information for all nodes.

#### W.2.1.1.2 Moab RM Language Resources Response Format

The query resources response format is one or more line of the following format (separated with a new line):

```
<NODEID><ATTR>=<VALUE>[ ; <ATTR>=<VALUE>] . . .
```

*<ATTR>* is one of the names in the [table below](#) and the format of *<VALUE>* is dependent on *<ATTR>*.

**Example 3-184: Moab RM language resource query and response****Request:**

```
CMD=GETNODES ARG=0:node001:node002:node003
```

**Response:**

```
node001 UPDATETIME=963004212;STATE=Busy;OS=AIX43;ARCH=RS6000...
node002 UPDATETIME=963004213;STATE=Busy;OS=AIX43;ARCH=RS6000...
...
```

**W.2.1.2 Moab RM Language Query Workload****W.2.1.2.1 Moab RM Language Query Workload Request Format**

```
CMD=GETJOBS ARG={<UPDATETIME>:<JOBID>[:<JOBID>]... | <UPDATETIME>:ALL }
```

Only jobs updated more recently than *<UPDATETIME>* will be returned where *<UPDATETIME>* is specified as the epoch time of interest. Setting *<UPDATETIME>* to 0 will return information for all jobs. Specify a colon delimited list of *JOBID*'s if information for specific jobs is desired or use the keyword *ALL* to receive information about all jobs.

**W.2.1.2.2 Moab RM Language Query Workload Response Format**

```
SC=<STATUSCODE> ARG=<JOBCOUNT>#<JOBID>:<FIELD>=<VALUE>; [ <FIELD>=<VALUE>; ] ...
[ #<JOBID>:<FIELD>=<VALUE>; [ <FIELD>=<VALUE>; ] ... ] ...
```

or

```
SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*FIELD* is either the text name listed below or A<*FIELDNUM*>  
(ie, UPDATETIME or A2)

*STATUSCODE* values:

- 0 SUCCESS
- -1 INTERNAL ERROR

*RESPONSE* is a statuscode sensitive message describing error or state details.

**W.2.1.2.3 Moab RM Language Query Workload Example****Request:**

```
CMD=GETJOBS ARG=0:ALL
```

**Response:**

```
ARG=2#nebo3001.0:UPDATETIME=9780000320;STATE=Idle;WCLIMIT=3600;...
```

### W.2.1.3 StartJob

The `StartJob` command may only be applied to jobs in the `Idle` state. It causes the job to begin running using the resources listed in the `NodeID` list.

```
send CMD=STARTJOB ARG=<JOBID> TASKLIST=<NODEID>[:<NODEID>]...
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

`STATUSCODE`  $\geq 0$  indicates SUCCESS

`STATUSCODE`  $< 0$  indicates FAILURE

`RESPONSE` is a text message possibly further describing an error or state

*Example 3-185: Job start*

```
# Start job nebo.1 on nodes cluster001 and cluster002
# send
CMD=STARTJOB ARG=nebo.1 TASKLIST=cluster001:cluster002
# receive
SC=0;RESPONSE=job nebo.1 started with 2 tasks
```

### W.2.1.4 CancelJob

The `CancelJob` command, if applied to an active job, will terminate its execution. If applied to an idle or active job, the `CancelJob` command will change the job's state to `Canceled`.

```
send CMD=CANCELJOB ARG=<JOBID> TYPE=<CANCELTYPE>
```

`<CANCELTYPE>` is one of the following:

ADMIN (command initiated by scheduler administrator)

WALLCLOCK (command initiated by scheduler because job exceeded its specified wallclock limit)

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

`STATUSCODE`  $\geq 0$  indicates SUCCESS

`STATUSCODE`  $< 0$  indicates FAILURE

`RESPONSE` is a text message further describing an error or state

*Example 3-186: Job cancel*

```
# Cancel job nebo.2
# send
CMD=CANCELJOB ARG=nebo.2 TYPE=ADMIN'
# receive
SC=0 RESPONSE=job nebo.2 canceled
```

### W.2.1.5 SuspendJob

The `SuspendJob` command can only be issued against a job in the state `Running`. This command [suspends](#) job execution and results in the job changing to the `Suspended` [state](#).

```
send CMD=SUSPENDJOB ARG=<JOBID>
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE* >= 0 indicates SUCCESS

*STATUSCODE* < 0 indicates FAILURE

*RESPONSE* is a text message possibly further describing an error or state

*Example 3-187: Job suspend*

```
# Suspend job nebo.3
# send
CMD=SUSPENDJOB ARG=nebo.3
# receive
SC=0 RESPONSE=job nebo.3 suspended
```

### W.2.1.6 ResumeJob

The `ResumeJob` command can only be issued against a job in the state `Suspended`. This command resumes a suspended job returning it to the `Running` state.

```
send CMD=RESUMEJOB ARG=<JOBID>
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE* >= 0 indicates SUCCESS

*STATUSCODE* < 0 indicates FAILURE

*RESPONSE* is a text message further describing an error or state

*Example 3-188: Job resume*

```
# Resume job nebo.3
# send
CMD=RESUMEJOB ARG=nebo.3
# receive
SC=0 RESPONSE=job nebo.3 resumed
```

### W.2.1.7 RequeueJob

The `RequeueJob` command can only be issued against an active job in the state `Starting` or `Running`. This command [requeues](#) the job, stopping execution and returning the job to an idle [state](#) in the queue. The queued job will be eligible for execution the next time resources are available.

```
send CMD=REQUEUEJOB ARG=<JOBID>
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE* >= 0 indicates SUCCESS

*STATUSCODE* < 0 indicates FAILURE

*RESPONSE* is a text message further describing an error or state

**Example 3-189: job requeue**

```
# Requeue job nebo.3
# send
CMD=REQUEUEJOB ARG=nebo.3
# receive
SC=0 RESPONSE=job nebo.3 requeued
```

**W.2.1.8 SignalJob**

The `SignalJob` command can only be issued against an active job in the state `Starting` or `Running`. This command signals the job, sending the specified signal to the master process. The signaled job will remain in the same state it was before the signal was issued.

```
send CMD=SIGNALJOB ARG=<JOBID> ACTION=signal VALUE=<SIGNAL>
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE*  $\geq 0$  indicates SUCCESS

*STATUSCODE*  $< 0$  indicates FAILURE

*RESPONSE* is a text message further describing an error or state

**Example 3-190: Job signal**

```
# Signal job nebo.3
# send
CMD=SIGNALJOB ARG=nebo.3 ACTION=signal VALUE=13
# receive
SC=0 RESPONSE=job nebo.3 signaled
```

**W.2.1.9 ModifyJob**

The `ModifyJob` command can be issued against any active or queued job. This command modifies specified attributes of the job.

```
send CMD=MODIFYJOB ARG=<JOBID> [BANK=name] [NODES=num] [PARTITION=name]
[TIMELIMIT=minutes]
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE*  $\geq 0$  indicates SUCCESS

*STATUSCODE*  $< 0$  indicates FAILURE

*RESPONSE* is a text message further describing an error or state

**Example 3-191: Job modify**

```
# Signal job nebo.3
# send
CMD=MODIFYJOB ARG=nebo.3 TIMELIMIT=9600
# receive
SC=0 RESPONSE=job nebo.3 modified
```

### W.2.1.10 JobAddTask

The `JobAddTask` command allocates additional tasks to an active job.

```
send CMD=JOBADDTASK ARG=<JOBID> <NODEID> [<NODEID>] ...
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE*  $\geq 0$  indicates SUCCESS

*STATUSCODE*  $< 0$  indicates FAILURE

*RESPONSE* is a text message possibly further describing an error or state

*Example 3-192: Job addtask*

```
# Add 3 default tasks to job nebo30023.0 using resources located on nodes cluster002,
cluster016, and cluster112.

# send
CMD=JOBADDTASK ARG=nebo30023.0 DEFAULT cluster002 cluster016 cluster112
# receive
SC=0 RESPONSE=3 tasks added
```

### W.2.1.11 JobRemoveTask

The `JobRemoveTask` command removes tasks from an active job.

```
send CMD=JOBREMOVETASK ARG=<JOBID> <TASKID> [<TASKID>] ...
```

```
receive SC=<STATUSCODE> RESPONSE=<RESPONSE>
```

*STATUSCODE*  $\geq 0$  indicates SUCCESS

*STATUSCODE*  $< 0$  indicates FAILURE

*RESPONSE* is a text message further describing an error or state

*Example 3-193: Job removetask*

```
# Free resources allocated to tasks 14, 15, and 16 of job nebo30023.0

# send
CMD=JOBREMOVETASK ARG=nebo30023.0 14 15 16
# receive
SC=0 RESPONSE=3 tasks removed
```

## W.2.2 Rejection Codes

- 0xx - success - no error
  - 00x - success
    - 000 - success
  - 01x - usage/help reply
    - 010 - usage/help reply
  - 02x - status reply
    - 020 - general status reply

- 1xx - warning
  - 10x - general warning
    - 100 - general warning
  - 11x - no content
    - 110 - general wire protocol or network warning
    - 112 - redirect
    - 114 - protocol warning
  - 12x - no matching results
    - 120 - general message format warning
    - 122 - incomplete specification (best guess action/response applied)
  - 13x - security warning
    - 130 - general security warning
    - 132 - insecure request
    - 134 - insufficient privileges (response was censored/action reduced in scope)
  - 14x - content or action warning
    - 140 - general content/action warning
    - 142 - no content (server has processed the request but there is no data to be returned)
    - 144 - no action (no object to act upon)
    - 146 - partial content
    - 148 - partial action
  - 15x - component defined
  - 18x - application defined
- 2xx - wire protocol/network failure
  - 20x - protocol failure
    - 200 - general protocol/network failure
  - 21x - network failure
    - 210 - general network failure
    - 212 - cannot resolve host
    - 214 - cannot resolve port

- 216 - cannot create socket
- 218 - cannot bind socket
- 22x - connection failure
  - 220 - general connection failure
  - 222 - cannot connect to service
  - 224 - cannot send data
  - 226 - cannot receive data
- 23x - connection rejected
  - 230 - general connection failure
  - 232 - connection timed-out
  - 234 - connection rejected - too busy
  - 236 - connection rejected - message too big
- 24x - malformed framing
  - 240 - general framing failure
  - 242 - malformed framing protocol
  - 244 - invalid message size
  - 246 - unexpected end of file
- 25x - component defined
- 28x - application defined
- 3xx - messaging format error
  - 30x - general messaging format error
    - 300 - general messaging format error
  - 31x - malformed XML document
    - 310 - general malformed XML error
  - 32x - XML schema validation error
    - 320 - general XML schema validation
  - 33x - general syntax error in request
    - 330 - general syntax error in response
    - 332 - object incorrectly specified
    - 334 - action incorrectly specified
    - 336 - option/parameter incorrectly specified

- 34x - general syntax error in response
  - 340 - general response syntax error
  - 342 - object incorrectly specified
  - 344 - action incorrectly specified
  - 346 - option/parameter incorrectly specified
- 35x - synchronization failure
  - 350 - general synchronization failure
  - 352 - request identifier is not unique
  - 354 - request id values do not match
  - 356 - request id count does not match
- 4xx - security error occurred
  - 40x - authentication failure - client signature
    - 400 - general client signature failure
    - 402 - invalid authentication type
    - 404 - cannot generate security token key - inadequate information
    - 406 - cannot canonicalize request
    - 408 - cannot sign request
  - 41x - negotiation failure
    - 410 - general negotiation failure
    - 412 - negotiation request malformed
    - 414 - negotiation request not understood
    - 416 - negotiation request not supported
  - 42x - authentication failure
    - 420 - general authentication failure
    - 422 - client signature failure
    - 424 - server authentication failure
    - 426 - server signature failure
    - 428 - client authentication failure
  - 43x - encryption failure
    - 430 - general encryption failure
    - 432 - client encryption failure

- 434 - server decryption failure
  - 436 - server encryption failure
  - 438 - client decryption failure
- 44x - authorization failure
  - 440 - general authorization failure
  - 442 - client authorization failure
  - 444 - server authorization failure
- 45x - component defined failure
- 48x - application defined failure
- 5xx - event management request failure
  - 50x - reserved
    - 500 - reserved
- 6xx - reserved for future use
  - 60x - reserved
    - 600 - reserved
- 7xx - server side error occurred
  - 70x - server side error
    - 700 - general server side error
  - 71x - server does not support requested function
    - 710 - server does not support requested function
  - 72x - internal server error
    - 720 - general internal server error
  - 73x - resource unavailable
    - 730 - general resource unavailable error
    - 732 - software resource unavailable error
    - 734 - hardware resource unavailable error
  - 74x - request violates policy
    - 740 - general policy violation
  - 75x - component-defined failure
  - 78x - application-defined failure

- 8xx - client side error occurred
  - 80x - general client side error
    - 800 - general client side error
  - 81x - request not supported
    - 810 - request not supported
  - 82x - application specific failure
    - 820 - general application specific failure
- 9xx - miscellaneous
  - 90x - general miscellaneous error
    - 900 - general miscellaneous error
  - 91x - general insufficient resources error
    - 910 - general insufficient resources error
  - 99x - general unknown error
    - 999 - unknown error

Related topics

- [Moab Resource Manager Language Data Format](#)
- [Managing Resources Directly with the Native Interface](#)

### W.3 Moab RM Language Socket Protocol Description

Moab RM language is formerly known as WIKI. The Moab scheduler uses a simple protocol for socket connections to the user client and the resource manager as described below:

`<SIZE><CHAR>CK=<CKSUM><WS>TS=<TIMESTAMP><WS>AUTH=<AUTH><WS>DT=<DATA>`

Attribute	Description
<b>&lt;SIZE&gt;</b>	8 character decimal ASCII representation of the size of the packet following ' <code>&lt;SIZE&gt;&lt;CHAR&gt;</code> ' Leading zeroes must be used to pad this value to 8 characters if necessary.
<b>&lt;CHAR&gt;</b>	A single ASCII character
<b>&lt;CKSUM&gt;</b>	A 16 character hexadecimal ASCII DES-based checksum calculated using the algorithm below* and <code>&lt;SEED&gt;</code> selected and kept secret by the site admins. The checksum is performed on the line from <code>TS=</code> to the end of the message including <code>&lt;DATA&gt;</code> .
<b>&lt;WS&gt;</b>	a series of white space characters consisting of either tabs and/or space characters.

Attribute	Description
<TIMESTAMP>	ASCII representation of epoch time
<AUTH>	Identifier of user requesting service (i.e., USERNAME)
<DT>	Data to be sent

An example header follows:

```
00001057 CK=cdf6d7a7ad45026f TS=922401962 AUTH=sched DT=<DATA>
```

where <DATA> is replaced by actual message data.

### W.3.1 Checksum Algorithm ('C' version)

```

#define MAX_CKSUM_ITERATION 4

int GetChecksum(
    char *Buf,
    int BufSize,
    char *Checksum,
    char *CSKey) /* Note: pass in secret key */
{
    unsigned int crc;
    unsigned int lword;
    unsigned int irword;
    int index;
    unsigned int Seed;
    Seed = (unsigned int) strtoul(CSKey, NULL, 0);
    crc = 0;
    for (index = 0; index < BufSize; index++)
    {
        crc = (unsigned int) DoCRC((unsigned short) crc, Buf[index]);
    }
    lword = crc;
    irword = Seed;
    PSDES(&lword, &irword);
    sprintf(Checksum, "%08x%08x",
        lword,
        irword);
    return(SUCCESS);
}

unsigned short DoCRC(
    unsigned short crc,
    unsigned char onech)
{
    int index;
    unsigned int ans;
    ans = (crc ^ onech << 8);
    for (index = 0; index < 8; index++)
    {
        if (ans & 0x8000)
            ans = (ans <<= 1) ^ 4129;
        else
            ans <<= 1;
    }
    return((unsigned short) ans);
}

int PSDES(
    unsigned int *lword,
    unsigned int *irword)
{
    int index;
    unsigned int ia;
    unsigned int ib;
    unsigned int iswap;
    unsigned int itmph;
    unsigned int itmpl;
    static unsigned int c1[MAX_CKSUM_ITERATION] = {
        0xcba4e531, 0x537158eb, 0x145cdc3c, 0x0d3fdeb2 };
    static unsigned int c2[MAX_CKSUM_ITERATION] = {
        0x12be4590, 0xab54ce58, 0x6954c7a6, 0x15a2ca46 };
    itmph = 0;

```

```
itmpl = 0;
for (index = 0; index < MAX_CKSUM_ITERATION; index++)
{
    iswap = *irword;
    ia = iswap ^ c1[index];
    itmpl = ia & 0xffff;
    itmph = ia >> 16;
    ib = (itmpl * itmpl) + ~(itmph*itmph);
    ia = (ib >> 16) | ((ib & 0xffff) << 16);
    *irword = (*lword) ^ ((ia ^ c2[index]) + (itmpl * itmph));
    *lword = iswap;
}
return(SUCCESS);
}
```

### W.3.2 Header Creation (PERL code)

(taken from PNNL's QBank client code)

```
#####
#
# subroutine wiki($COMMAND)
#
# Sends command to Moab server and returns the parsed result and status
#
#####
sub wiki
{
  my($COMMAND,$REQUEST,$result);
  my($sockaddr,$hostname);
  my($name,$aliases,$proto,$port,$type,$len,$thisaddr);
  my($thisport,$thatport,$response,$result);
  $COMMAND = shift;
  #
  # Establish socket connection
  #
  $sockaddr = 'S n a4 x8';
  chop ($hostname = `hostname`);
  ($name,$aliases,$proto)=getprotobyname('tcp');
  ($name,$aliases,$type,$len,$thisaddr)=gethostbyname($hostname);
  ($name,$aliases,$type,$len,$thataddr)=gethostbyname($BANKHOST);
  $thisport=pack($sockaddr, &AF_INET,0,$thisaddr);
  $thatport=pack($sockaddr, &AF_INET,$BANKPORT,$thataddr);
  socket(S, &PF_INET,&SOCK_STREAM,$proto) || die "cannot create socket\n";
  bind(S,$thisport) || die "cannot bind socket\n";
  connect(S,$thatport) || die "cannot connect socket\n";
  select(S); $| = 1; # Turn on autoflushing
  select(stdout); $| = 1; # Select STDOUT as default output
  #
  # Build and send command
  #
  $REQUEST="COMMAND=$COMMAND AUTH=$AUTH";
  chomp($CHECKSUM = `QSUM "$REQUEST"`);
  $REQUEST .= " CHECKSUM=$CHECKSUM";
  my $command=pack "a8 a1 A*",sprintf("%08d",length($REQUEST))," ",$REQUEST;
  print S "$command"; # Send Command to server
  @REPLY=();
  while () { push(@REPLY,$_); } # Listen for Reply

  $STATUS=grep(/STATUSCODE=(\d*)/,&&$1,@REPLY); # STATUSCODE stored in $STATUS
  grep(s/.*RESULT=//,@REPLY); # Parse out the RESULT
  return @REPLY;

}

```

### W.3.3 Header Processing (PERL code)

```

sysread(NS,$length,8); # Read length string
sysread(NS,$delimiter,1); # Read delimiter byte
$DEBUG && print STDERR "length=[$length]\tdelimiter=[$delimiter]\n";
while($length) {
  $DEBUG && print STDERR "Awaiting $length bytes -- ".`date`;
  $length-=sysread(NS,$request,$length); # Read request
  sleep 1;
}
%REQUEST=();
chomp($request);
foreach (@REQUEST=&shellwords($request)) # Parse arguments into array
{
  ($key,$value)=split(/=/,$_);
  $REQUEST{$key}=$value unless defined $REQUEST{$key};
}
$request =~ s/\s+CHECKSUM=.*//; # Strip off the checksum
print STDERR "REQUEST=$request\n";
chomp($checksum=`$QSUM "$request"`);
$me=$REQUEST{AUTH};
$command=$REQUEST{COMMAND};
if (!grep($command eq $_,@VALIDCMDS))
{ $REPLY = "STATUSCODE=0 RESULT=$command is not a valid command\n";}
elsif ($checksum ne $REQUEST{CHECKSUM})
{ $REPLY = "STATUSCODE=0 RESULT=Invalid Checksum\n";}
else
{ $REPLY = do $command(@REQUEST); }
$len=sprintf("%08d",length($REPLY)-1);
$delim=' ';
$DEBUG && print STDERR "REPLY=${len}${delim}$REPLY\n";
$buf="$len"."$delim"."$REPLY";
syswrite(NS,$buf,length($buf));
close NS;

```

## SCHEDCFG flags

Flag	Description
<b>AGGREGATENODEFEATURES</b>	<i>AGGREGATENODEFEATURES</i> causes Moab to aggregate features reported by the different RMs. For example, if you have two RMs reporting different features for the same node, Moab will add both features together (instead of one being overwritten by the other). In order to set features manually, you can use <code>mnodectl -m features</code> (for details, see <a href="#">mnodectl on page 229</a> ).
<b>ALLOWINFINITEJOBS</b>	<i>ALLOWINFINITEJOBS</i> allows infinite wallclock times to be accepted. Previously, jobs with infinite job times were allowed by default.

Flag	Description
<b>ALLOWMULTICOMPUTE</b>	<i>ALLOWMULTICOMPUTE</i> tells Moab how to resolve conflicting information from different resource managers. If <i>ALLOWMULTICOMPUTE</i> is specified, Moab will use the STATE and OS information from the resource manager that reports the node as online.
<b>CANCELFAILEDDEPENDENCYJOBS</b>	Automatically cancels dependency jobs that will never run because of an unmet requirement. For example, if you ran a job with both an afterok and afternotok job attached to it and that job was successful, the afterok job would run, leaving the afternotok job idle in the queue. If you set <i>CANCELFAILEDDEPENDENCYJOBS</i> , Moab will cancel the job with the failed dependency and remove it from the queue. For more information about job dependencies, see <a href="#">Job Dependencies on page 528</a> .
<b>DISABLEPERJOBNODESETS</b>	Disables a job's ability to override the system specified node set. See <a href="#">13.3 Resource Manager Extensions</a> for more information.
<b>DISABLEPARTIALNODERESERVATIONS</b>	Blocks partial node reservations.
<b>ENABLESLURMMEMPERCPU</b>	By default Moab calls sbatch or srun with a --mem= request in a SLURM environment. When you set <i>ENABLESLURMMEMPERCPU</i> , Moab instead calls --mem-per-cpu=. This is to allow sites with policies that require the other parameter to use --mem-per-cpu.
<b>ENFORCERESERVEDNODES</b>	Without this flag Moab tries to optimize the reservation for a job before it starts, meaning a job may start on nodes that weren't part of its reservation. With this flag Moab tries to start jobs only on the nodes that were reserved.
<b>ENFORCESAMENODESET</b>	The same nodeset is not enforced across job requirements by default, rather each requirement is scheduled separately and the nodesets are determined on a per-req basis. To have Moab enforce the same nodeset across all job requirements set this flag.
<b>EXTENDEDGROUPSUPPORT</b>	Allows Moab to consider a user's secondary Linux groups when dealing with reservation <a href="#">ACLs</a> .
<b>FASTGROUPLOOKUP</b>	Moab will use the system call <code>getgrouplist</code> to gather group information. This can significantly improve performance on some LDAP systems.

Flag	Description
<b>FASTRSVSTARTUP</b>	<p>Speeds up start time if there are existing reservations.</p> <div data-bbox="716 359 1432 447" style="border: 1px solid #ccc; padding: 5px; margin-bottom: 10px;">  <b>FASTRSVSTARTUP</b> is incompatible with partial node reservations. </div> <p>On very large systems, if there is a reservation in the checkpoint file on all the nodes, it would take a really long time for Moab to start up. For every node in the reservation, Moab checks every other node. With this flag, Moab just uses the nodelist that was checkpointed to create the reservation. It speeds up the startup process because it doesn't have to check every node. Where Moab would take 8 - 10 minutes to start up with an 18,000 node reservation without the flag, Moab can start up in 2-3 minutes with the flag.</p> <p>With the flag you will see one difference in <a href="#">checknode</a>. A reservation that uses all the procs on a node initially shows that all the procs are blocked. Without the flag, and as jobs fill on the node, the blocked resources will be configured - dedicated (ex. 5/6). With the flag, the blocked resources will always be what the reservation is blocking and won't change when jobs fill on the node.</p> <p>Without flag:  Reservations:  brian.1x1 User -00:12:52 -&gt; INFINITY ( INFINITY)  Blocked Resources@-00:00:02 Procs: 5/6 (83.33%) Mem: 0/5000 (0.00%)  Blocked Resources@00:04:58 Procs: 6/6 (100.00%) Mem: 0/5000 (0.00%)  m.2x1 Job:Running -00:00:02 -&gt; 00:04:58 (00:05:00)  Jobs: m.2</p> <p>With flag:  Reservations:  brian.1x1 User -00:00:15 -&gt; INFINITY ( INFINITY)  Blocked Resources@-00:00:02 Procs: 6/6 (100.00%) Mem: 0/5000 (0.00%)  Blocked Resources@00:04:58 Procs: 6/6 (100.00%) Mem: 0/5000 (0.00%)  m.1x1 Job:Running -00:00:02 -&gt; 00:04:58 (00:05:00)  Jobs: m.1</p> <div data-bbox="716 1539 1432 1627" style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  When you set the FASTRVSTARTUP flag, Moab will also set the <a href="#">DISABLEPARTIALNODERESERVATIONS</a> flag. </div>
<u><a href="#">FILELOCKHA</a></u>	<p>This is a <a href="#">High Availability</a> feature. <b>FILELOCKHA</b> prevents scheduling conflicts between multiple Moab servers.</p>

Flag	Description
<b>FREECOMPLETEDJOBSSUBMITSTRING</b>	Moab frees the job submit string for completed jobs, decreasing the amount of memory needed during operation. This is useful in environments with large job scripts that can create a large memory footprint.
<b>IGNOREPIDFILELOCK</b>	Moab will not fail if it cannot get a lock on the <code>.moab.pid</code> file. This is useful when Moab is running on a shared filesystem where file locking can be unpredictable.
<b>JOBSUSERSVWALLTIME</b>	Allows jobs submitted without a walltime request or default walltime received from a class or queue but with an <code>ADVRES:reservation</code> to inherit their walltime limit from the reservation instead of the Moab default. The job walltime limit is then the remaining time of the reservation to which the job was submitted.
<b>NOCLASSUPDATE</b>	While running against TORQUE, Moab will not update classes when it refreshes each iteration. Moab loads the classes at startup, but does not refresh them until the next time it is restarted.
<b>NORMALIZETASKDEFINITIONS</b>	<p>Instructs Moab to normalize all tasks that it receives via an <a href="#">mshow -a</a> command. Moab normalizes the task definition to one processor and then changes the tasks requested to the number of processors requested. For example, when the following is received by Moab:</p> <pre>mshow -a -w mintasks=1@procs:4+mem:4096</pre> <p>It is changed to this:</p> <pre>mshow -a -w mintasks=4@procs:1+,mem:1024,tpn=4</pre>
<b>OPTIMIZEDBACKFILL</b>	On large systems that utilize system-wide reservations, backfill can take a considerable amount of time. This flag speeds up backfill scheduling by using an alternative BETA backfill algorithm. This flag will be the default in future versions of Moab.

Flag	Description
<b>PRIORITYPOLICYBLOCKING</b>	<p>By default, a job that violates a policy is placed into the blocked queue. Jobs with a lower priority, but that do not violate the policy, will run. This can lead to situations in which small jobs starve out larger, higher priority jobs.</p> <p>When you set the <i>PRIORITYPOLICYBLOCKING</i> flag, Moab allows the job that violates the policy to continue consuming the policy slots while it remains blocked. With the policy slots consumed, the smaller, lower priority jobs will not run. The higher priority job will continue to consume the policy slots until it has consumed enough to actually run.</p> <p>Note that because the blocked job consumes policy slots, this will inevitably lead to lower system utilization.</p>
<b>SHOWREQUESTEDPROCS</b>	<p>Shows requested processors regardless of NodeAccessPolicy in <a href="#">showq</a>. When <i>SINGLEJOB NODEACCESSPOLICY</i> is used and the job requests one processor, <a href="#">showq</a> displays the job with one processor.</p>
<b>SHOWUSERJOBONLY</b>	<p>Causes Moab, when a non-admin user runs <a href="#">showq</a>, to return only that user's jobs. If an administrator runs <a href="#">showq</a> when this flag is set, Moab returns the jobs of all users; no restrictions are placed on administrators.</p>
<b>STRICTSPOOLDIRPERMISSIONS</b>	<p>Enforces at least a 511 permission on the Moab spool directory.</p>
<b>UNMIGRATEONDEFER</b>	<p>Forces Moab to unmigrate a job in a grid if it enters a deferred state.</p>



# Moab Web Services

## Moab Web Services overview

Moab Web Services (MWS) is a component of Adaptive Computing Suites that enables programmatic interaction with Moab Workload Manager via a RESTful interface. MWS lets you create and interact with Moab objects and properties such as jobs, nodes, virtual machines, and reservations. MWS is the preferred method for those wishing to create custom user interfaces for Moab and is the primary method by which Moab Viewpoint communicates with Moab.

MWS communicates with the Moab Workload Manager (Moab) server using the same wire protocol as the Moab command-line interface. By publishing a standard interface into Moab's intelligence, MWS significantly reduces the amount of work required to integrate Moab into your solution.

This documentation is intended for developers performing such integrations. If you are a Moab administrator, and for conceptual information about Moab, see [Moab Workload Manager overview on page 89](#).

## Setup

### Moab Web Services setup

This section explains what you need to know in order to get MWS configured, and secured correctly. It contains the following topics:

- [Configuring Moab Web Services on page 1395](#)
- [Setting up MWS security on page 1410](#)
- [Version and build information on page 1418](#)

Related topics

- [Moab Web Services overview on page 1395](#)
- [Access control on page 1420](#)

### Configuring Moab Web Services

This section describes the location of the MWS configuration files. It also shows some examples of how to configure logging.

**i** To see a full reference to all configuration and logging parameters available in MWS, see [Configuration on page 1772](#).

This topic contains these sections:

- [Home directory on page 1396](#)
- [Configuration files on page 1396](#)
- [Logging configuration using mws-config.groovy on page 1396](#)
- [LDAP Configuration using mws-config.groovy on page 1402](#)
- [PAM \(pluggable authentication module\) configuration using mws-config.groovy on page 1405](#)
- [OAuth configuration using mws-config.groovy on page 1407](#)

## Home directory

The MWS home directory contains configuration files, log files, and files that serve features of MWS such as hooks and plugins. You should set the location of the MWS home directory using the **MWS\_HOME** property. If you do not set **MWS\_HOME** as a Java property or as an environment variable, then MWS will use `/opt/mws` as the default **MWS\_HOME**.

## Configuration files

The primary configuration file is `MWS_HOME/etc/mws-config.groovy`. If this file is missing or contains errors, MWS will not start.

Configuration files can also be placed in the `MWS_HOME/etc/mws.d` directory. Any configuration files here get merged with `MWS_HOME/etc/mws-config.groovy`. In case of conflict, the configuration in `MWS_HOME/etc/mws.d` takes precedence.

If `MWS_HOME/etc/log4j.properties` exists, MWS will load it as well.

## Logging configuration using mws-config.groovy

Shown below is an example that logs all error messages and fatal messages to `/opt/mws/log/mws.log` (For information about the format of the MWS logs, see "[Standard Log Format](#)" in the *Moab Workload Manager Administrator Guide*). It also logs all stack traces to `/opt/mws/log/stacktrace.log`. Note that this example is not configured to log events; for details on logging events, see [Configuring an event log on page 1397](#).

## Minimal logging configuration

```

log4j = {
  appenders {
    rollingFile name: 'stacktrace',
      file: '/opt/mws/log/stacktrace.log',
      maxFileSize: '1GB'
    rollingFile name: 'rootLog',
      file: '/opt/mws/log/mws.log',
      threshold: org.apache.log4j.Level.ERROR,
      maxFileSize: '1GB'
  }
  root {
    debug 'rootLog'
  }
}

```

Alternatively, you may configure a console appender instead of a rolling file, as shown below.

## Console logging configuration

```

log4j = {
  appenders {
    rollingFile name: 'stacktrace',
      file: '/opt/mws/log/stacktrace.log',
      maxFileSize: '1GB'
    console name: 'consoleLog',
      threshold: org.apache.log4j.Level.ERROR
  }
  root {
    debug 'consoleLog'
  }
}

```



You may configure logging by using either `MWS_HOME/etc/mws-config.groovy` or `MWS_HOME/etc/log4j.properties`.

If you do not define any `log4j` configuration, MWS will write its log files to `java.io.tmpdir`. For Tomcat, `java.io.tmpdir` is generally set to `$CATALINA_BASE/temp` or `CATALINA_TMPDIR`.

## Configuring an event log

Logging events to a flat file requires that you make a few changes to the configuration in the **log4j** section of the `mws-config.groovy` file so that events will be logged to the `events.log` file, and all other MWS logging information will be sent to the `mws.log` file.

### Causing events.log to roll based on a time window

You can specify how often the `events.log` file rolls. The following example illustrates the configuration changes you will need make to `mws-config.groovy` to cause the `events.log` file to roll based on a time window. Note the following three examples:

- In this example, `mws-config.groovy` is configured so that `events.log` rolls daily at midnight.

```

Daily rolling events.log configuration in mws-config.groovy
-----

log4j = {
  def eventAppender = new org.apache.log4j.rolling.RollingFileAppender(name:
'events', layout: pattern(conversionPattern: "%m%n"))
  def rollingPolicy = new org.apache.log4j.rolling.TimeBasedRollingPolicy
(fileNamePattern: '/tmp/events.%d{yyyy-MM-dd}', activeFileName:
'/tmp/events.log')
  rollingPolicy.activateOptions()
  eventAppender.setRollingPolicy(rollingPolicy)

  appenders {
    appender eventAppender

    rollingFile name: 'rootLog',
      file: '/tmp/mws.log',
      maxFileSize: '1GB'
  }

  root {
    warn 'rootLog'
  }

  trace additivity:false, events:'com.ace.mws.events.EventFlatFileWriter'
}

```

Note the **RollingFileAppender** and the **TimeBasedRollingPolicy** lines. These lines configure MWS to write the event log to the `events.log` file. Rolled log files will have a date appended to their name in this format: "yyyy-MM-dd" (for example, `events.log.2012-02-28`).

- If you want the event log file to roll at the beginning of each month, change the **fileNamePattern** `TimeBasedRollingPolicy` date format to `yyyy-MM`. For example:

```

Monthly event logs
-----

def rollingPolicy = new org.apache.log4j.rolling.TimeBasedRollingPolicy
(fileNamePattern: '/tmp/events.%d{yyyy-MM}', activeFileName: '/tmp/events.log')

```

- If you want the event log file to roll at the beginning of each hour, change the date format to `yyyy-MM-dd_HH:00`. For example:

```

Hourly event logs
-----

def rollingPolicy = new org.apache.log4j.rolling.TimeBasedRollingPolicy
(fileNamePattern: '/tmp/events.%d{yyyy-MM-dd_HH:00}', activeFileName:
'/tmp/events.log')

```

### Configuring events.log to roll based on a file size threshold

You can also configure the `events.log` file to roll when the log size exceeds a specified threshold. The following example illustrates the configuration changes you will need to make to `mws-config.groovy` to cause the `events.log` file to roll on a size threshold. (In this example, `mws-config.groovy` is configured so that `events.log` rolls when its size exceeds 50 MB.)

```
mws-config.groovy configuration that rolls events.log based on file size
```

```
-----
log4j = {
  appenders {
    rollingFile name: 'events',
      file: '/tmp/events.log',
      maxFileSize: '50MB',
      maxBackupIndex:10

    rollingFile name: 'rootLog',
      file: '/tmp/mws.log',
      maxFileSize: '1GB'
  }

  root {
    warn 'rootLog'
  }

  trace additivity:false, events:'com.ace.mws.events.EventFlatFileWriter'
}

```

Note that **maxFileSize** is set to "50MB." This means that when the `events.log` file exceeds 50 MB, it will roll.

The name for the rolled log will be "events.log.1". When the *new* `events.log` file exceeds 50 MB, *it* will roll and be named "events.log.1", while the old "events.log.1" file will be renamed "events.log.2". This process will continue until the optional **maxBackupIndex** value is met. In the example above, **maxBackIndex** is set to 10. This means that MWS will delete all but the ten most recent `events.log` files. Using this feature helps prevent hard drives from filling up.

### Additivity

The **additivity** attribute of the `EventFlatFileWriter` logger can be either `true` or `false`. If you specify `true`, events will be logged to the `events.log` file *and* the `mws.log` file. If you specify `false`, events will be logged to the `events.log` file only. (All other MWS logging information will be logged to the `mws.log` file, as configured by the `rootLog` appender.)

To log events to the `mws.log` file in addition to the `events.log` file, make the `additivity:true` configuration. For example:

```
-----
Logging events to both events.log and mws.log

trace additivity:true, events:'com.ace.mws.events.EventFlatFileWriter'
```

For more configuration options, see [Apache Extras Companion for log4j](#).

### Deleting old events

If your MongoDB server is version 2.2 or later, MongoDB will automatically delete events older than 30 days (by default). For more information, including how to change this default, see `mws.events.expireAfterSeconds` in [Configuration on page 1772](#).

If your MongoDB server is older than version 2.2, MongoDB will store event data indefinitely. However, if disk space is limited, you may want to regularly delete old, unneeded events from MongoDB. This section contains some examples of how you can do this.

Let's say that you want to delete events that are older than 90 days. (There are 86,400,000 milliseconds in a day, so in this example,  $90 \times 86400000$  corresponds to 90 days in milliseconds.):

- You could run this script:

```

Delete events older than 90 days
-----
$ mongo
MongoDB shell version: 2.4.8
connecting to: test
> use mws
> db.event.remove({eventTime:{$lt:new Date(new Date()).getTime()-90*86400000}})
> exit

```

- To create a script to perform this task:

```

deleteOldEvents.sh
-----
#!/bin/bash
printf 'use mws_dev\ndb.event.remove({eventTime:{$lt:new Date(new Date()).getTime()-90*86400000}})\nexit' | mongo

```

- Now say that you want to set up a [cron job](#) (`crontab -e`) so that old events are automatically deleted on a certain day of the week (for example, every Sunday at 2:00 a.m.), you would add an entry like this:

```

cron table entry to delete old events
-----
00 02 * * 0 /root/deleteOldEvents.sh

```

## Configuring an audit trail log

Audit logging enables you to track changes to [Permissions on page 1593](#), [Roles on page 1655](#), and [Principals on page 1627](#).



Sample audit.log format:

```
Audit trail log format
-----
2013-10-30 14:39:32,120 TENANT 'admin' updated resource named 'Engineering2' with
values:
  "name": "Engineering3",
  "attachedPrincipals": [{"name": "Engineering"}]
```

## LDAP Configuration using `mws-config.groovy`

### Using a supported LDAP directory type

To configure an MWS connection to an LDAP server, add the following parameters to `mws-config.groovy`:

**i** Throughout the following examples in this topic, you will see **dc=acme,dc=com**. "acme" is only used as an example to illustrate what you would use as your own domain controller if your domain name was "acme.com." You should replace any references to "acme" with your own organization's domain name.

Parameter	Description
<b>ldap.server</b>	The hostname or IP address of the LDAP server.
<b>ldap.port</b>	The port the LDAP server is listening on.
<b>ldap.baseDNs</b>	A list of distinguished names that are the root entries for LDAP searches.
<b>ldap.bindUser</b>	The distinguished name of the bind user.
<b>ldap.password</b>	The password of the ldap.bindUser.
<b>ldap.directory.type</b>	The type of LDAP directory (e.g. "Microsoft Active Directory"). This parameter can have the following values: <ul style="list-style-type: none"> <li>• Microsoft Active Directory</li> <li>• OpenLDAP Using InetOrgPerson Schema</li> <li>• OpenLDAP Using NIS Schema</li> <li>• OpenLDAP Using Samba Schema</li> </ul>

Here is a sample configuration for OpenLDAP.

**i** If you followed the Adaptive Computing tutorial [\[link\]"Setting up OpenLDAP on CentOS 6"](#) your **ldap.directory.type** should be set to "OpenLDAP Using InetOrgPerson Schema".

## Sample OpenLDAP configuration

```

ldap.server = "192.168.0.5"
ldap.port = 389
ldap.baseDNs = ["dc=acme,dc=com"]
ldap.bindUser = "cn=Manager,dc=acme,dc=com"
ldap.password = "*****"
ldap.directory.type = "OpenLDAP Using InetOrgPerson Schema"

```

Here is a sample configuration for Microsoft Active Directory.

## Sample Active Directory configuration

```

ldap.server = "192.168.0.5"
ldap.port = 389
ldap.baseDNs = ["CN=Users,DC=acme,DC=com", "OU=Europe,DC=acme,DC=com"]
ldap.bindUser = "cn=Administrator,cn=Users,DC=acme,DC=com"
ldap.password = "*****"
ldap.directory.type = "Microsoft Active Directory"

```



To see how to configure a secure connection to the LDAP server, see [Securing the LDAP connection on page 1416](#).

## Using an unsupported LDAP directory type

If you are not using one of the supported directory types, you can explicitly configure MWS to work with your LDAP schema by using the following parameters:

Parameter	Description
<b>ldap.user.objectClass</b>	The name of the class used for the LDAP user object. For example: <ul style="list-style-type: none"> <li>• user</li> <li>• person</li> <li>• inetOrgPerson</li> <li>• posixAccount</li> </ul>
<b>ldap.group.objectClass</b>	The name of the class used for the LDAP group object. For example: <ul style="list-style-type: none"> <li>• group</li> <li>• groupOfNames</li> <li>• posixGroup</li> </ul>
<b>ldap.ou.objectClass</b>	The name of the class used for the LDAP organizational unit object. for example: <ul style="list-style-type: none"> <li>• organizationalUnit</li> </ul>

Parameter	Description
<b>ldap.user.membership.attribute</b>	The attribute field in a user entry to use when loading the user's groups (optional if <b>ldap.group.membership.attribute</b> is defined). For example: <ul style="list-style-type: none"> <li>• memberOf</li> </ul>
<b>ldap.group.membership.attribute</b>	The attribute field in a group entry to use when loading the group's members (optional if <b>ldap.user.membership.attribute</b> is defined). For example: <ul style="list-style-type: none"> <li>• member</li> <li>• memberUid</li> </ul>
<b>ldap.user.name.attribute</b>	The attribute field to use when loading the username. This field must uniquely identify a user. For example: <ul style="list-style-type: none"> <li>• sAMAccountName</li> <li>• uid</li> </ul>

For example:

```
Advanced Active Directory configuration
-----
ldap.server = "myldaphostname"
ldap.port = 389
ldap.baseDNs = ["CN=Users,DC=acme,DC=com", "OU=Europe,DC=acme,DC=com"]
ldap.bindUser = "cn=Administrator,cn=Users,DC=acme,DC=com"
ldap.password = "*****"
ldap.user.objectClass = "person"
ldap.group.objectClass = "group"
ldap.ou.objectClass = "organizationalUnit"
ldap.user.membership.attribute = "memberof"
ldap.group.membership.attribute = "member"
ldap.user.name.attribute = "sAMAccountName"
```

Here is a similar example for OpenLDAP. Note there is no user membership attribute in the OpenLDAP InetOrgPerson schema and thus **ldap.user.membership.attribute** is set to null. This is allowable because the **ldap.group.membership.attribute** is set.

```
Advanced OpenLDAP configuration
-----
ldap.server = "myldaphostname"
ldap.port = 389
ldap.baseDNs = ["dc=acme,dc=com"]
ldap.bindUser = "cn=Manager,dc=acme,dc=com"
ldap.password = "*****"
ldap.user.objectClass = "inetOrgPerson"
ldap.group.objectClass = "groupOfNames"
ldap.ou.objectClass = "organizationalUnit"
ldap.user.membership.attribute = null
ldap.group.membership.attribute = "memberUid"
ldap.user.name.attribute = "uid"
```

## Overriding attributes in a supported LDAP directory type

You can also override attributes in supported directory types. For example, say you are using OpenLDAP with an NIS Schema. The group objectClass for NIS defaults to "groupOfNames," but you want to use "groupOfUniqueNames" instead while retaining all other defaults for NIS. You can do this by setting **ldap.directory.type** to "OpenLDAP Using NIS Schema" and overriding the **ldap.group.objectClass** attribute as follows:

```
Advanced OpenLDAP configuration
-----
ldap.directory.type = "OpenLDAP Using NIS Schema"
ldap.group.objectClass = "groupOfUniqueNames"
```

**i** LDAP is *not* currently used to authenticate users to MWS. LDAP is only used to map principals to roles, as explained in [Principals on page 1627](#).

The user class in your LDAP schema must have an attribute that uniquely identifies a user (for example: "uid" or "sAMAccountName").

## PAM (pluggable authentication module) configuration using `mws-config.groovy`

PAM functions as bridge to the underlying Unix authentication system. PAM treats the user as if it is local to the Unix machine doing the authenticating and uses whatever the Unix user is authenticating with, whether it be LDAP or NIS. PAM uses configuration files that specify the how, when, or what for authentication, session management, and account management. Each configuration file can be different. For example, `sudo` configuration file for the "sudo" command will handle authentication differently than the `login` configuration file. These configuration files are dynamically read for `/etc/pam.d`.

### Requirements for PAM

In order to use PAM with MWS, the following is required:

- The PAM application package must be installed. For example:

```
yum install pam
```

- You must have a PAM configuration file in the `/etc/pam.d` directory. The following is an example of what a PAM configuration file might look like:

```

#%PAM-1.0
auth        required      pam_env.so
auth        sufficient    pam_unix.so nullok try_first_pass
auth        requisite     pam_succeed_if.so uid >= 1000 quiet_success
auth        required      pam_deny.so

account     required      pam_unix.so
account     sufficient    pam_localuser.so
account     sufficient    pam_succeed_if.so uid < 1000 quiet
account     required      pam_permit.so

password    requisite     pam_pwquality.so try_first_pass retry=3 authtok_type=
password    sufficient    pam_unix.so sha512 shadow nullok try_first_pass use_
authtok
password    required      pam_deny.so

session     optional      pam_keyinit.so revoke
session     required     pam_limits.so
-session    optional     pam_systemd.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet
use_uid
session     required     pam_unix.so
    
```

- (Optional) You must have PAM modules installed for your specific needs.

The PAM application comes with default modules—for example, **pam\_unix.xo**—that will check username and password credentials with Unix. You may have to install others for your distribution.

### Configuring MWS to use PAM

To configure an MWS connection to PAM, add the following parameter to `mws-config.groovy`:

Parameter	Description
<b>pam.configuration.service</b>	The name of the PAM configuration file located in <code>/etc/pam.d</code> . This parameter and specification tells MWS which PAM configuration file you want to use.

For example:

```
pam.configuration.service = "system-auth"
```

**i** You can configure only one authentication method in `mws-config.groovy`—LDAP or PAM, but not both. If you have configured both LDAP and PAM, MWS defaults to using LDAP.

If you need multiple authentication methods, you must add them to your local PAM configuration. See your distribution documentation for details.



There is a security risk when authenticating local users through your PAM configuration. This behavior is highly discouraged and not supported by Adaptive Computing.

For more information about PAM, please see the following [SLES](#) and [RedHat](#) documentation.

## OAuth configuration using `mws-config.groovy`

OAuth is a security framework designed to simplify authentication in web technologies. In the case of MWS, OAuth allows trusted client applications to securely delegate authentication to MWS. Once MWS has authenticated a user by verifying the username and password in LDAP, PAM, or NIS, MWS returns an access token to the client. The client then presents this access token to MWS to access resources. OAuth is very flexible and allows MWS to work in many different scenarios by use of grant types. For more information on OAuth and grant types, please see the following [OAuth](#) documentation.

### Example using 'password' grant type

#### [Terminology](#)

**Resource Owner:** The person accessing and manipulating data. For MWS, this would be the person who logs into the client (the user).

**Service Provider:** The site or service where protected resources live. This can be (but is not necessarily) also the identify provider, where usernames and passwords are stored. This is the MWS service itself.

**Client:** The application that wants to access a resource. For MWS this is the user interface, potentially including APIs and command-line tools.

**Protected Resource:** The data for which protection is desired. For MWS this would be Moab itself, and interaction with Moab.

**Access Token:** Instead of user credentials, OAuth uses tokens to issue requests, and the tokens get signed to indicate authorization.

### Register a client in MWS

OAuth requires client registration. Its client credentials are used to validate that the client is allowed to authenticate on behalf of a resource owner. It involves giving the client its own credentials (username and password). MWS will first authenticate the client using a client id (username) and client secret (password), then will authenticate the resource owner.

Add the following line to `/opt/mws/etc/mws-config.groovy`:

```
grails.plugin.springsecurity.oauthProvider.clients = [
  [
    clientId:"THE_CLIENT_ID",
    clientSecret:"THE_CLIENT_SECRET",
    authorizedGrantTypes:["password"]
  ]
]
```

Replace `THE_CLIENT_ID` with client id (username). For example: `clientId:"iris"`. Also, replace `THE_CLIENT_SECRET` with client secret (password). For example: `clientSecret:"irisclientpassword"`,. Note that the values for `clientId` and `clientSecret` are case sensitive.

You can register more than one client. For example:

```
grails.plugin.springsecurity.oauthProvider.clients = [
  [
    clientId:"client_id_1",
    clientSecret:"client_secret_1",
    authorizedGrantTypes:["password"]
  ],
  [
    clientId:"client_id_2",
    clientSecret:"client_secret_1",
    authorizedGrantTypes:["password"]
  ]
]
```

### Obtain an access token from MWS for a resource owner (logging in)

Before the client can access private data in MWS, the client must obtain an access token that grants access to the API. The token endpoint url is only used to gain an access token and log in a user.

Getting an access token:

```
POST http://localhost:8080/mws/rest/oauth/token?api-version=3
Adding header:
  "Content-Type: application/x-www-form-urlencoded"
Request body (String):
grant_type=password&client_id=THE_CLIENT_ID&client_secret=THE_CLIENT_SECRET&username=RESOURCE_OWNER_USERNAME&password=RESOURCE_OWNER_PASSWORD
```

Example using curl:

```
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" -v -d 'grant_type=password&client_id=iris&client_secret=irisclientpassword&username=moab-admin&password=secret' 'http://localhost:8080/mws/oauth/token'
```

Produces the following response:

```

* About to connect() to localhost port 8080 (#0)
*   Trying 127.0.0.1... connected
* Connected to localhost (127.0.0.1) port 8080 (#0)
> POST /mws/oauth/token HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0
zlib/1.2.3 libidn/1.18 libssh2/1.4.2
> Host: localhost:8080
> Accept: */*
> Content-Type: application/x-www-form-urlencoded
> Content-Length: 126
>
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Cache-Control: no-store
< Pragma: no-cache
< Set-Cookie: JSESSIONID=6CE8F9E7C454575FABCF3D156B153CFD; Path=/mws
< Content-Type: application/json;charset=UTF-8
< Transfer-Encoding: chunked
< Date: Fri, 18 May 2014 18:16:42 GMT
<
* Connection #0 to host localhost left intact
* Closing connection #0
{"access_token":"b693eec0-6c93-4540-8b2f-1e170be08046","token_type":"bearer","expires_in":43096}

```

### Send the access token to MWS when requesting protected resource

After the client obtains an access token, it will send the access token to MWS in an HTTP authorization header for each rest call.

 The client is responsible for handling user sessions with each access token, meaning the client has to request a new access token when a new user logs in.

Requesting an MWS resource (getting list of all nodes for example):

```

GET http://localhost:8080/mws/rest/nodes?api-version=3&fields=name
Adding authorization header:
    "Authorization: Bearer ACCESS_TOKEN"

```

Example using curl:

```

curl -X GET -H "Authorization: Bearer b693eec0-6c93-4540-8b2f-1e170be08046" -v
'http://localhost:8080/mws/rest/nodes?api-version=3&fields=name'

```

Produces the following response:

```

* About to connect() to localhost port 8080 (#0)
* Trying 127.0.0.1... connected
* Connected to localhost (127.0.0.1) port 8080 (#0)
> GET /mws/rest/nodes?api-version=3&fields=name HTTP/1.1
> User-Agent: curl/7.19.7 (x86_64-redhat-linux-gnu) libcurl/7.19.7 NSS/3.14.0.0
zlib/1.2.3 libidn/1.18 libssh2/1.4.2
> Host: localhost:8080
> Accept: */*
> Authorization: Bearer b693eec0-6c93-4540-8b2f-1e170be08046
>
< HTTP/1.1 200 OK
< Server: Apache-Coyote/1.1
< Content-Type: application/json;charset=UTF-8
< Pragma: no-cache
< Set-Cookie: JSESSIONID=6CE8F9E7C454575FABCF3D156B153CFD; Path=/mws
< Content-Type: application/json;charset=UTF-8
< Content-Language: en-US
< Transfer-Encoding: chunked
< Date: Fri, 18 May 2014 18:39:07 GMT
<
{"totalCount":3,"resultCount":3,"results":[{"name":"node1"}, {"name":"node2"},
{"name":"node3"}]}

```

### Related topics

- [Setting up MWS security on page 1410](#)
- [Version and build information on page 1418](#)

## Setting up MWS security

When running MWS in production environments, security is a major concern. This section focuses on securing the connections with MWS:

- The connection between MWS and Moab Workload Manager (see [Securing the connection with Moab on page 1411](#)).
- The connection between MWS and MongoDB (see [Securing the connection with MongoDB on page 1411](#)).
- The connections between clients and MWS (see [Securing client connections to MWS on page 1412](#)).
- The connection between MWS and LDAP (see [Securing the LDAP connection on page 1416](#)).
- The connection with the message queue (see [Securing the connection with the message queue on page 1417](#)).

### Related topics

- [Configuring Moab Web Services on page 1395](#)
- [Version and build information on page 1418](#)

## Securing the connection with Moab

MWS communicates with Moab via the Moab Wire Protocol, which uses a direct connection between the two applications. The communication over this connection uses a shared secret key, which is discussed in the installation instructions (see "[Installing Moab Web Services on page 31](#)" in the installation documentation). However, the communication is not encrypted and is therefore susceptible to eavesdropping and replay attacks. For this reason, MWS is supported only when running on the same machine as Moab. This assures that any connections between the two applications occur internally on the server and are not exposed to external users.

### Related topics

- [Setting up MWS security on page 1410](#)

## Securing the connection with MongoDB

By default, the connection between MWS and MongoDB is not authenticated. To enable authentication, follow the instructions below. For further reading, see the MongoDB tutorial "[Control Access to MongoDB Instances with Authentication](#)."

### To enable an authenticated connection between MWS and MongoDB

1. Add an administrative user to the `admin` database.
2. Add an MWS user to the `mws` database.
3. To support MWS API version 2, add an MWS user with "read-only" rights to the `moab` database.

Here is an example of how to create all the required users. The users in the `moab` database are required only for MWS API version 2.

```
[root]# service mongod start
[root]# mongo
> use admin;
> db.addUser("admin_user", "secret1");
> use moab;
> db.addUser("moab_user", "secret2");
> db.addUser("mws_user", "secret3", true);
> use mws;
> db.addUser("mws_user", "secret3");
> exit;
```

**i** The passwords used here ("secret1," "secret2," and "secret3") are examples. Choose your own passwords for these users.

4. Add the MWS user credentials (the ones you just created) to the `mws-config.groovy` file. For example:

```
grails.mongo.username = "mws_user"
grails.mongo.password = "secret3"
```

5. Enable authentication in the MongoDB configuration file (called `/etc/mongodb.conf` on many Linux distributions). In that file, look for `#auth = true` and uncomment it.
6. Restart MongoDB.
7. Restart Tomcat.

If authentication is enabled in MongoDB, but the MWS user was not properly created or configured, MWS will not start. In this case, see the log file(s) for additional information.

#### Related topics

- [Setting up MWS security on page 1410](#)

## Securing client connections to MWS

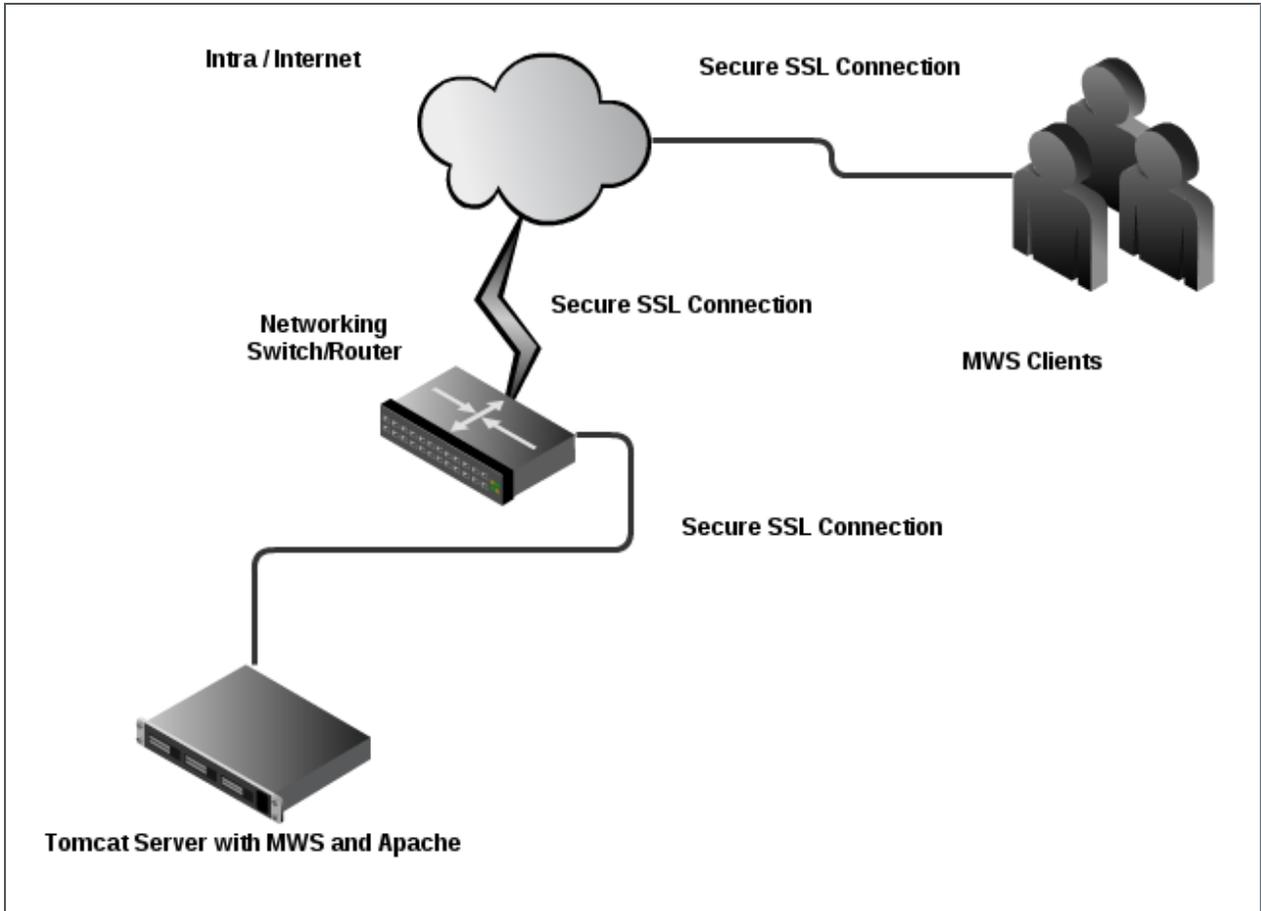
All connections to MWS, except those requesting the documentation or the main page, must be authenticated properly. MWS uses a single-trusted-user authentication model, meaning a single user exists that has access to all aspects of MWS. The username and password for this user are configured with the `auth.defaultUser` properties in the configuration file. For more information, see [Configuration on page 1772](#).

When using the MWS user interface in a browser, the user will be prompted for username and password. For information on how to authenticate requests when not using a browser, see [Authentication on page 1443](#).

 The username and password in the Basic Authentication header are encoded but not encrypted. Therefore, it is *strongly* recommended that MWS be run behind a proxy (like Apache) with SSL enabled. The instructions below provide an example of how to do this.

## Encrypting client connections using Apache and SSL

This section shows how to encrypt client connections to MWS using Apache and SSL. These instructions have been tested on CentOS™ 6.2 with the "Web Server" software set installed. The same ideas are applicable to other operating systems, but the details might be different. As shown in the diagram below, these instructions assume that Tomcat and Apache are running on the same server.



### To encrypt client connections using Apache and SSL

1. Create a self-signed certificate. (If desired, see <http://www.openssl.org/docs/HOWTO/certificates.txt> for more information.)

**i** Instead of creating a self-signed certificate, you can buy a certificate from a certificate vendor. If you do, then the vendor will provide instructions on how to configure Apache with your certificate.

2. Do the following:
  - a. Run these commands:

```
cd /etc/pki/tls/certs
cp -p make-dummy-cert make-dummy-cert.bak
cp -p localhost.crt localhost.crt.bak
```

- b. Edit `make-dummy-cert` and replace the `answers()` function with code similar to this:

```

answers() {
  echo US
  echo Utah
  echo Provo
  echo Adaptive Computing Enterprises, Inc.
  echo Engineering
  echo test1.adaptivecomputing.com
  echo
}

```

- c. Run this command:

```
./make-dummy-cert localhost.crt
```

3. Configure Apache to use the new certificate and to redirect MWS requests to Tomcat. To do so, edit `/etc/httpd/conf.d/ssl.conf`. Do the following"

- a. Comment out this line:

```
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
```

- b. Add these lines near the end, just above `</VirtualHost>`:

```
ProxyPass /mws http://127.0.0.1:8080/mws retry=5
ProxyPassReverse /mws http://127.0.0.1:8080/mws
```

4. Configure Apache to use SSL for all MWS requests. Add these lines to the end of `/etc/httpd/conf/httpd.conf`:

```
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (/mws.*) https://%{HTTP_HOST}%{REQUEST_URI}
```

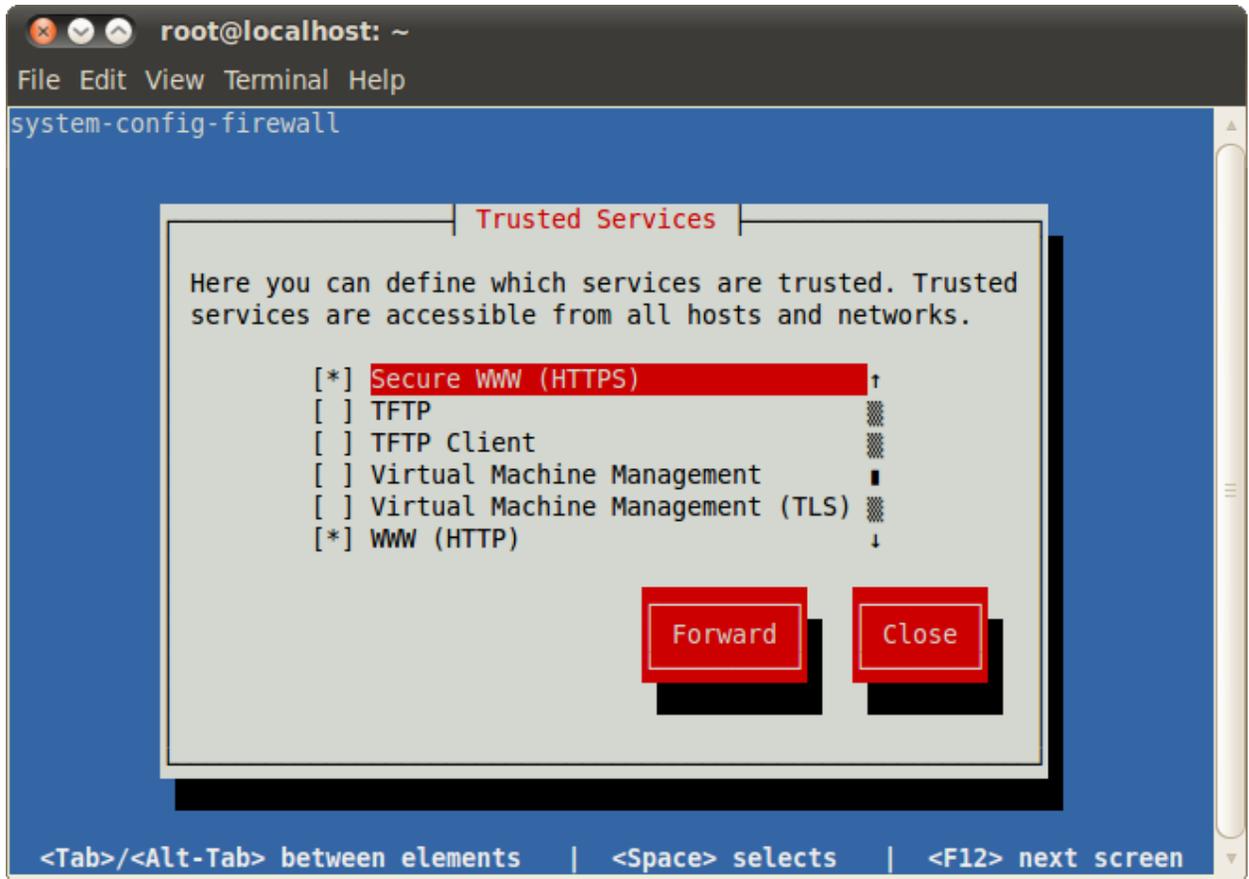
5. Give Apache permission to connect to Tomcat.

```
setsebool -P httpd_can_network_connect 1
```

6. Turn on Apache.

```
chkconfig httpd on
service httpd start
```

7. Using `system-config-firewall-tui`, enable "Secure WWW (HTTPS)" and "WWW (HTTP)" as trusted services.



## Encrypting client connections using Tomcat and SSL

This section shows how to encrypt client connections to MWS using Tomcat and SSL but without requiring the use of Apache. These instructions have been tested on CentOS™ 6.2 with Tomcat 6.0.

### To encrypt client connections using Tomcat and SSL

1. First, you must generate a certificate. Do the following:
  - a. Use the keytool utility that is shipped with the Oracle Java Runtime Environment. As the Tomcat user, run the following:
 

```
keytool -genkey -alias tomcat -keyalg RSA
```
  - b. Specify a password value of "changeit". This will create a .keystore file that contains the new certificate in the user's home directory.
2. Enable the Tomcat SSL connector. Do the following:
  - a. Open the `server.xml` file, usually located in `$CATALINA_HOME/conf/` (`$CATALINA_HOME` represents the directory where Tomcat is installed).

- b. Verify the SSL HTTP/1.1 Connector entry is enabled. To do so locate the SSL HTTP/1.1 Connector entry and uncomment it.

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true" maxThreads="150"
scheme="https" secure="true" clientAuth="false" sslProtocol="TLS" />
```

**i** The code above enables SSL access on port 8443. The default for HTTPS is 443, but just as Tomcat uses 8080 instead of 80 to avoid conflicts, 8443 is used instead of 443.

- c. Save the `server.xml` file.
- d. Verify that `server.xml` is owned by the Tomcat user.

```
chown -R tomcat:tomcat server.xml
```

- e. Next modify the `MWSweb.xml` file. Add a security-constraint section to the `$_CATALINA_HOME/webapps/mws/WEB-INF/web.xml` file found in your Tomcat directory.

```
<web-app>
...
  <security-constraint>
    <web-resource-collection>
      <web-resource-name>MWS Secure URLs</web-resource-name>
      <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <user-data-constraint>
      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
  </security-constraint>
</web-app>
```

- f. Now restart tomcat.

## Related topics

- [Setting up MWS security on page 1410](#)

## Securing the LDAP connection

All connections from MWS to the LDAP server should be secured with SSL or StartTLS to ensure passwords and other sensitive information are encrypted as they pass to and from the LDAP server. If the LDAP server does not support SSL or StartTLS, the rest of this section is irrelevant.

### Determine whether the LDAP server's certificate is trusted

If the LDAP server's X.509 certificate has been signed by a trusted certificate authority such as Verisign, Thawte, GeoTrust, and so on, Java will trust the certificate automatically and you won't need to add the certificate to Java's keystore. Consult your IT department to determine whether the LDAP server certificate has been signed by a trusted certificate authority. If the LDAP server certificate is signed by a trusted certificate authority, skip ahead to [Configure MWS to connect to LDAP server using SSL or](#)

[StartTLS on page 1417](#). Otherwise, follow the instructions in [Trusting servers in Java on page 77](#) to add the certificate to Java's keystore.

## Configure MWS to connect to LDAP server using SSL or StartTLS

### To configure MWS to connect to LDAP using SSL/TLS

1. Update the `ldap.port` and `ldap.security.type` parameters in `/opt/mws/etc/mws-config.groovy`.

```
ldap.port = 636
ldap.security.type = "SSL"
```

### To configure MWS to connect to LDAP using StartTLS

1. Update the `ldap.port` and `ldap.security.type` parameters in `/opt/mws/etc/mws-config.groovy`.

```
ldap.port = 389
ldap.security.type = "StartTLS"
```

The table below lists the possible values for `ldap.security.type`:

ldap.security.type	Default port	Notes
<b>None</b>	389	This is the default if no security type is configured. All data is sent in plain text.
<b>SSL</b>	636	Requires server certificate. All data is encrypted.
<b>StartTLS</b>	389	Starts as an insecure connection and is upgraded to an SSL/TLS connection. Requires server certificate. After upgrade all data is encrypted.

## Securing the connection with the message queue

MWS supports message queue security with AES. If the `moab.messageQueue.secretKey` property is set, then all messages MWS publishes on the message queue will be encrypted. Additionally, MWS can read messages from Moab Workload Manager that are encrypted with the same key using the `MESSAGEQUEUESECRETKEY` parameter. For more information, see [Configuration on page 1772](#).

Encryption is done with AES in CBC mode where inputs are padded with PKCS5 padding. Only 128-bit (16-byte) keys are supported. Keys should be encoded in [Base64](#).

For example:

```
moab.messageQueue.secretKey = "1r6RvfqJa6voezy5wAx0hw==" //must be a Base64-encoded 128-bit key
```

**i** Important: If MWS is configured to encrypt the message queue and Moab is not (or vice versa) then the messages from Moab will be ignored. Furthermore, all attempts to access the MWS service resource will fail.

### Related topics

- [Resources introduction](#) on page 1446
- [Events](#) on page 1528
- [Notifications](#) on page 1585
- [Notification conditions](#) on page 1580
- [Creating events and notifications](#) on page 1704
- [Plugin developer's guide](#) on page 1679
- [Fields: Events](#) on page 1894
- [Plugin event service](#) on page 1755
- [Handling events](#) on page 1711
- [System events](#) on page 1444
- [Securing the connection with the message queue](#) on page 1417

## Version and build information

To get detailed version information about MWS, use one of the following three methods:

- [Browser](#) on page 1418
- [REST request](#) on page 1419
- [MANIFEST.MF file](#) on page 1419

### Browser

Using a browser, visit the MWS home page (for example, <http://localhost:8080/mws/>). At the bottom of the page is the MWS version information. See the screenshot below:

**Migrate a VM:**

VM:

```
{ "node": { "id": "hv1" } }
```

Moab Web Services 7.0.0-beta-3, Build 993 (2012-02-04\_16-15-33), Revision 79f9da5b00e8a36e5cf40b5c96b61a04e9813fe9

**REST request**

Using a REST client or other HTTP client software, send a GET request to the `rest/diag/about` resource. Here is an example:

```
curl -u username:password http://localhost:8080/mws/rest/diag/about?api-version=3
```

This resource is also described under [Diagnostics on page 1521](#).

**MANIFEST.MF file**

If MWS fails to start, version and build information can be found in the `META-INF/MANIFEST.MF` file inside the MWS WAR file. The version properties begin with **Implementation**. Below is an excerpt of a `MANIFEST.MF` file:

```
Implementation-Build: 26
Implementation-Build-Date: 2012-06-19 14-18-59
Implementation-Revision: 376079a5e5f552f2fe25e6070fd2e84c646a98fd

Name: Grails Application
Implementation-Title: mws
Implementation-Version: 7.1.0-rc2
Grails-Version: 2.0.3
```

**Related topics**

- [Setting up MWS security on page 1410](#)

# Access control

## About access control

### Access control

This section describes how to manage access control in MWS. Applications are the consumers of MWS. They include Moab Viewpoint and other applications that need the resources provided by MWS. An application account consists of four editable fields and resource-specific access control settings:

Table 4-1: Field information

Field	Required	Default value	Value type	Maximum length	Description
<b>Application Name</b>	Yes	--	String	32	The name of the application. Must start with a letter and may contain letters, digits, underscores, periods, hyphens, apostrophes, and spaces.
<b>Username</b>	Yes	--	String	32	Used for authentication. Must start with a letter and may contain letters, digits, underscores, periods, and hyphens.
<b>Description</b>	No	--	String	1000	The description of the application.
<b>Enabled</b>	--	true	Boolean	--	Controls whether the application is allowed to access MWS.
<b>Access Control Settings</b>	Yes	All Per-missions	--	--	The permissions granted to the application. This is controlled by selecting specific check boxes in a grid.

An application account also contains an auto-generated password that is visible only when creating the account or when resetting its password. Whenever an application sends a REST request to MWS, it needs to pass its credentials (username and password) in a Basic Authentication header. For more information, see [Authentication on page 1443](#).

The **Application Name** is a human-friendly way to identify an application account, but MWS does not use it during authentication (or at any other time, for that matter).

The **Enabled** field is set to true automatically when an application account is created. To change the value of this field, see [Modifying an application account on page 1422](#).

Here is an example of how you might set the fields when creating an application account:

- **Application Name:** Moab Viewpoint
- **Username:** viewpoint
- **Description:** This application account grants access to Moab Viewpoint for Moab Cloud Suite.

The permissions granted to an application account may be customized while creating or modifying the account. For more information, see [Creating an application account on page 1421](#) and [Modifying an application account on page 1422](#).

## Managing application accounts

Application accounts are used to grant access to MWS. Every application with an application account must be granted at least one access control permission to a resource in MWS. To manage application accounts, see [Listing application accounts on page 1421](#).

## Listing application accounts

To list all applications accounts, browse to the MWS home page (for example, <https://servername/mws>). Log in as the admin user, click **Admin** and then **Application Accounts**.

Each column (except **Password**) can be sorted in ascending or descending order by clicking on the column heading.

## Creating an application account

To create an application account, go to the **Application List** page and click **Add Application**. The "Application Name" and "Username" are required fields. For more details, see [Field information on page 1420](#).

Access to specific resources and plugin custom web services is granted or revoked by checking or unchecking the check boxes in the respective resources or plugin web services access control sections. For each resource, access may be granted to a resource for each method supported by MWS, including GET, POST, PUT, and DELETE. See the figure below for an example.

<input type="checkbox"/> Select All	<input checked="" type="checkbox"/> GET	<input type="checkbox"/> POST	<input checked="" type="checkbox"/> PUT	<input checked="" type="checkbox"/> DELETE
<input checked="" type="checkbox"/> <b>Access Control Lists</b>			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
<input checked="" type="checkbox"/> <b>Accounts</b>	<input checked="" type="checkbox"/>			
<input type="checkbox"/> <b>Events</b>	<input checked="" type="checkbox"/>	<input type="checkbox"/>		

In this example, the application has access to all available methods for the **Access Control Lists** and **Accounts** resources as well as to retrieve the **Events** resource through the GET method, but is denied the permission to create new events through the POST method.

Access may also be granted to each plugin type's custom web service(s). When new plugin types or plugin web services are added to MWS, applications must be updated with the new access control settings. See below for an example.

Plugin Type	<input checked="" type="checkbox"/> Can Access
<input checked="" type="checkbox"/> <b>Test</b>	
customService	<input checked="" type="checkbox"/>
unsecuredService (Unsecured)	

In this example, the application has access to all the custom web services defined for the **Test** plugin type. Note that though unsecured web services are listed, access to them cannot be denied (for more information, see [Exposing web services on page 1696](#)).

## Displaying an application account

To show information about an application account, go to the **Application List** page and click the desired application name.

In addition to displaying the values for fields, grids are also displayed which represent the application's access control permissions defined for resources and plugin custom web services. Examples of the resources and the plugin web services access control displays are shown below:

	GET	POST	PUT	DELETE
<b>Access Control Lists</b>				
<b>Accounts</b>				
<b>Events</b>				

Plugin Type	Can Access
<b>Test</b>	
customService	
unsecuredService (Unsecured)	

## Modifying an application account

To modify an application account, go to the **Application List** page, click the desired application name, and then click **Edit**. See [Creating an application account on page 1421](#) for more information on available fields and access control settings.

## Resetting an application password

To reset an application password, go to the **Application List** page and click the **Reset** link for the desired application. Alternatively, go to the **Display Application** page for the desired application and click the **Reset** link.

## Deleting an application account

To delete an application account, go to the **Application List** page, click the desired application name, and then click **Delete**. A confirmation message is shown. If the **OK** button is clicked, the application account is deleted from the system and cannot be recovered.

Related topics

- [Moab Web Services overview on page 1395](#)
- [Setting up MWS security on page 1410](#)

# API documentation

## About the API

Moab Web Services provide a set of RESTful resources that can be used to create, read, update, and delete various objects in the Moab Workload Manager. This section describes how to use RESTful web services, explains the JSON data format used for all communications with MWS, describes global URL parameters used in MWS calls, and contains other helpful information for using the Moab Web Services API.

This section contains these topics:

- [RESTful web services on page 1423](#)
- [Data format on page 1425](#)
- [Global URL parameters on page 1425](#)
- [Requesting specific API versions on page 1428](#)
- [Responses and return codes on page 1429](#)
- [Error messages on page 1432](#)
- [Pre and post-processing hooks on page 1434](#)
- [Authentication on page 1443](#)

Related topics

- [Resources introduction on page 1446](#)
- [About Moab Web Services plugins on page 1672](#)

## RESTful web services

In order to understand how to use MWS, it is first necessary to give a brief introduction to REST. REST (Representational State Transfer) is a set of guidelines which utilizes the full HTTP (Hypertext Transfer

Protocol) specification along with endpoint URLs that describe resources. The HTTP methods used in REST are comprised of the following:

Method	Description
<b>GET</b>	Query for a list or a single resource.
<b>POST</b>	Creating a resource.
<b>PUT</b>	Modifying a resource.
<b>DELETE</b>	Deleting a resource.

In comparison to other architectures of web services which use a single HTTP method and service endpoint to perform multiple types of operations (such as a POST operation to a URL), REST utilizes all of the available HTTP methods and URLs that directly correlate to resources. For example, RESTful web services for books in a library may expose many URL endpoints and the HTTP methods available for each such as GET, POST, PUT, and DELETE. The list below gives the methods, URLs, and descriptions for a sample set of services. The number 1 represents a unique identifier for books in each case.

Method	URL	Description
<b>GET</b>	/books	Retrieves a list of all books in the library.
<b>POST</b>	/books	Creates a new book.
<b>GET</b>	/books/1	Retrieves a single book.
<b>PUT</b>	/books/1	Modifies a single books.
<b>DELETE</b>	/books/1	Deletes a single book.

**i** Note that in the cases of the POST and PUT operations, additional information may be needed to describe the resource to be created or the fields that should be modified.

Moab Web Services provides RESTful web services for many resources. The methods and URLs available are documented in [Resources introduction on page 1446](#).

#### Related topics

- [About the API on page 1423](#)

## Data format

JSON (JavaScript Object Notation) is the data format used for all communication with MWS. This format makes use of two main structures: collections of key/value pairs called *objects* and ordered lists of values called *arrays*. Objects are defined by using curly braces ({}), and arrays are defined by using square brackets ([]). A JSON object or array may contain several different types of values including numbers, booleans (true/false), strings, objects, arrays, or the keyword 'null' representing no value. For example, a simple JSON object might be defined as:

```
{
  "number": 1,
  "decimalNumber": 1.2,
  "boolean": true,
  "string": "Any string",
  "dateString": "2013-05-23 17:32:02 UTC",
  "object": {
    "key": "value"
  },
  "array": [
    "value1",
    "value2"
  ],
  "nullValue": null
}
```

Dates in MWS, for both input and output, use the pattern "yyyy-MM-dd HH:mm:ss ZZZ". For more details on that pattern, see [Joda-Time DateTimeFormat](#). For a list of valid time zone IDs, see [Joda-Time Available Time Zones](#).

For more information on JSON, see [json.org](http://json.org).

The data format of MWS is defined as follows:

- Input for a POST or PUT must be in JSON format. Set the Content-Type header to application/json.
- Output is in JSON format and always consists of an object with zero or more key/value pairs.
- The output may also be "pretty-printed" or formatted for human viewing by sending a URL parameter. For more information, see [Global URL parameters on page 1425](#).

Related topics

- [About the API on page 1423](#)

## Global URL parameters

 All URL parameters are optional.

Parameter	Valid values	Description
<b>api-version</b>	Integer	Requests a specific API version
<b>pretty</b>	<i>true</i>	Controls pretty printing of output
<b>fields</b>	Comma-separated string	Includes only specified fields in output
<b>exclude-fields</b>	Comma-separated string	Excludes specified fields from output
<b>max</b>	Integer	The maximum number of items to return
<b>offset</b>	Integer	The index of the first item to return

## API version (api-version)

See [Requesting specific API versions on page 1428](#) for information on this parameter and how it should be used.

## Pretty (pretty)

By default, the output is easy for a machine to read but difficult for humans to read. The **pretty** parameter formats the output so that it is easier to read.

## Field selection (fields)

The **fields** parameter will include *only* the specified fields in the output. For list queries, the field selection acts on the objects in **results** and not on the **totalCount** or **results** properties themselves.

The format of the **fields** parameter is a comma-separated list of properties that should be included, as in `id, state`. Using periods, sub-objects may also be specified, and fields of these objects may be included as well. This is done with the same syntax for both single sub-objects and lists of sub-objects, as in `id, requirements.requiredNodeCountMinimum, blockReason.message`.

*Example 4-1: Example for a job query*

Request

```
GET /rest/jobs?api-
version=3&fields=name, flags, requirements.taskCount, dates.createdDate
```

## Response

```

-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "dates": {"createdDate": "2012-10-17 01:11:54 UTC"},
    "flags": ["GLOBALQUEUE"],
    "name": "Moab.24",
    "requirements": [{"taskCount": 1}]
  } ]
}

```

## Field exclusion (exclude-fields)

The **exclude-fields** parameter is the opposite of the **fields** parameter. All fields will be included in the output *except* those that are specified. For list queries, the field exclusion acts on the objects in **results** and not on the **totalCount** or **results** properties themselves.

The format of the **exclude-fields** parameter is a comma-separated list of properties that should be excluded from the output, as in `id,state`. Using periods, sub-objects may also be specified, and fields of these objects may be excluded as well. This is done with the same syntax for both single sub-objects and lists of sub-objects, as in

`id,requirements.requiredNodeCountMinimum,blockReason.message`.

### Example 4-2:

Suppose a query returns the following JSON:

## Request with No Field Exclusion

```
GET /objects
```

## Response

```

-----
{
  "id": "1",
  "listOfStrings": [
    "string1",
    "string2"
  ],
  "listOfObjects": [ {
    "item1": "value1",
    "item2": "value2"
  } ],
  "singleObject": {
    "id": "obj1",
    "field1": "value1"
  }
}

```

The same query with `exclude-fields` would return the following output:

```
Request with No Field Exclusion
```

```
GET /objects?exclude-fields=id,listOfObjects.item2,singleObject.field1,listOfStrings
```

```
Response
```

```
{
  "listOfObjects": [{"item1": "value1"}],
  "singleObject": {"id": "obj1"}
}
```

## Sorting (sort)

[Images on page 1536](#) and [Events on page 1528](#) support sorting based on [MongoDB syntax](#) by using the sort parameter. To sort in ascending order, specify a 1 for the sorting field. To sort in descending order, specify a -1. Objects can also be sorted on nested fields by using dot notation to separate the sub-fields, such as field.subfield1.subfield2.

### Related topics

- [About the API on page 1423](#)

## Requesting specific API versions

Because of significant changes in the API introduced in release version 7.2.0, MWS possesses a versioned API. The `api-version` URL parameter may be used to change the requested API version for any call to MWS. The current valid API versions with their corresponding MWS versions are shown in the table below:

API version	MWS version	Documentation	Additional notes
2 (deprecated)	7.2.x	7.2.x documentation on <a href="http://docs.adaptivecomputing.com/">http://docs.adaptivecomputing.com/</a>	As of the 8.0.0 release, API version 2 is <b>officially deprecated</b> and will be removed from Moab Web Services in the next major release.
3	8.0	Contained within this document	--
latest	Latest	Contained within this document	When the <code>latest</code> API version is requested, it resolves to the latest API version of MWS, such as <code>api-version=3</code> for MWS 8.0.0.



If no API version is specified, the request is rejected. An API version must be specified with every call in Moab Web Services 8.0.0 and later.

[Resources introduction on page 1446](#) and [Resources reference on page 1782](#) contain information for the latest API version. For documentation of previous API versions, please see the table above.

## Examples

```
GET http://localhost:8080/mws/rest/nodes?api-version=2
// Data returned uses API version 2

GET http://localhost:8080/mws/rest/nodes?api-version=latest
// Data returned uses API version 3
```

### Related topics

- [About the API on page 1423](#)

## Responses and return codes

Various HTTP responses and return codes are generated from MWS operations. These are documented below according to the operation that they are associated with.

- [Listing and showing resources on page 1429](#)
- [Creating resources on page 1430](#)
- [Modifying resources on page 1431](#)
- [Deleting resources on page 1431](#)
- [Moab response headers on page 1432](#)

### Listing and showing resources

For any successful list or show operation (GET), a 200 OK response code is always returned. No additional headers beyond those typical of a HTTP response are given in the response.

The body of this response consists of the results of the list or show operation. For a list operation, the results are wrapped in metadata giving total and result counts. The result count represents the number of resource records returned in the current request, and the total count represents the number of all records available. These differ when querying or the `max` and `offset` parameters are used. The following is an example of a list operation response:

## JSON List Response Body

```

{
  "resultCount":1,
  "totalCount":5,
  "results":[
    {
      "id":"Moab.1",
      ...
    }
  ]
}

```

For a show operation, the result is given as a single object:

## JSON Show Response Body

```

{
  "id":"Moab.1",
  ...
}

```

## Creating resources

A successful creation (POST) of a resource has two potential response codes:

- If the resource was created immediately, a 201 Created response code is returned.
- If the resource is still being created, a 202 Accepted response code is returned.

In either case, a `Location` header is added to the response with the full URL which can be used to get more information about the newly created resource or the task associated with creating the resource (if a 202 is returned).

Additionally, the body of the response will contain the unique identifier of the newly created resource or the unique identifier for the task associated with creating the resource (if a 202 is returned).

For example, during creation or submission of a job, a 201 response code is returned with the following response headers and body:

## Job Creation Response Headers

```

HTTP/1.1 201 Created
Server: Apache-Coyote/1.1
Location: /mws/rest/jobs/Moab.21
X-Moab-Status: Success
X-Moab-Code: 000
Content-Type: application/json;charset=utf-8
Content-Length: 16
Date: Wed, 21 Dec 2011 23:04:47 GMT

```

## Job Creation Response Body

```

{"id":"Moab.21"}

```

## Modifying resources

For any successful resource modification operation (`PUT`), a `200 OK` or `202 Accepted` response code is returned. A `200` response code signifies that the modification was immediately completed. No additional headers are returned in this case. A `202` response code is used again to signify that the modification is not yet complete and additional actions are taking place. In this case, a `Location` header is also returned with the full URL of the resource describing the additional actions.

In the case of a `200` response code, the body of this response typically consists of an object with a single `messages` property containing a list of statuses or results of the modification(s). However, a few exceptions to this rule exist as documented in [Resources introduction on page 1446](#). In the case of a `202` response code, the format is the same as for a `202` during a creation operation, in that the body consists of an object with the unique identifier for the task associated with the additional action(s).

For example, when modifying a job, several messages may be returned as follows with the associated `200` response code.

### Job Modification Response Headers

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
X-Moab-Status: Success
X-Moab-Code: 000
X-Moab-Message:
Content-Type: application/json;charset=utf-8
Content-Length: ...
Date: Thu, 22 Dec 2011 16:49:43 GMT
```

### JSON Modify Response Body

```
{
  "messages": [
    "gevent processed",
    "variables successfully modified"
  ]
}
```

## Deleting resources

For any successful resource deletion operation (`DELETE`), a `200 OK` or `202 Accepted` response code is returned. A `200` response code signifies that the deletion was immediately completed. No additional headers are returned in this case. A `202` response code is used again to signify that the deletion is not yet complete and additional actions are taking place. In this case, a `Location` header is also returned with the full URL of the resource describing the additional actions.

In the case of a `200` response code, the body of this response is empty. In the case of a `202` response code, the format is the same as for a `202` during a creation operation, in that the body consists of an object with the unique identifier for the task associated with the additional action(s).

For example, when deleting a job, a `200` response code is returned with an empty body as shown below.

```

Job Deletion Response
-----
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
X-Moab-Status: Success
X-Moab-Code: 000
X-Moab-Message:
Content-Type: application/json;charset=utf-8
Content-Length: 0
Date: Thu, 22 Dec 2011 16:49:43 GMT

```

## Moab response headers

In addition to the typical HTTP headers and the Location header described above, several headers are returned if the operations directly interact with Moab. These headers are described in the following table:

Name	Description
<b>X-Moab-Status</b>	One of Success, Warning, or Failure. Describes the overall status of the Moab request.
<b>X-Moab-Code</b>	A three digit code specifying the exact error encountered, used only in debugging.
<b>X-Moab-Message</b>	An optional message returned by Moab during the request.

### Related topics

- [About the API on page 1423](#)

## Error messages

Below is an explanation of what error message format to expect when an HTTP status code other than 20x is returned. All error codes have a response code of 400 or greater.

- [400 Bad request on page 1433](#)
- [401 Unauthorized on page 1433](#)
- [403 Forbidden on page 1433](#)
- [404 Not found on page 1433](#)
- [405 Method Not Allowed on page 1433](#)
- [500 Internal server error on page 1434](#)

## 400 Bad request

This response code is returned when the request itself is at fault, such as when trying to modify a resource with an empty `PUT` request body or when trying to create a new resource with invalid parameters. The response body is as follows:

```
{
  "messages": [
    "Message describing error",
    "Possible prompt to take action"
  ]
}
```

## 401 Unauthorized

This response code is returned when authentication credentials are not supplied or are invalid. The response body is as follows:

```
{
  "messages": [
    "You must be authenticated to access this area"
  ]
}
```

## 403 Forbidden

This response code is returned when the credentials supplied are valid, but the permissions granted are insufficient for the operation. This occurs when using application accounts (see [Access control on page 1420](#)) with limited access.

```
{
  "messages": [
    "You are not authorized to access this area"
  ]
}
```

## 404 Not found

This response code is returned when the request specifies a resource that does not exist. The response body is as follows:

```
{
  "messages": [
    "The resource with id 'uniqueId' was not found"
  ]
}
```

## 405 Method Not Allowed

This response code is returned when a resource does not support the specified HTTP method as an operation. The response body is as follows:

```
{
  "messages": [
    "The specified HTTP method is not allowed for the requested resource"
  ]
}
```

## 500 Internal server error

This indicates that there was an internal server error while performing the request, or that an operation failed in an unexpected manner. These are the most serious errors returned by MWS. If additional information is needed, the MWS log may contain further error data. The response body is as follows:

```
{
  "messages": [
    "A problem occurred while processing the request",
    "A message describing the error"
  ]
}
```

### Related topics

- [About the API on page 1423](#)

## Pre and post-processing hooks

MWS provides functionality to intercept and modify data sent to and returned from web services for all available resources. This is done by creating hooks in Groovy files located in a sub-directory of the `MWS_HOME` directory (by default, `/opt/mws/hooks`).



Please see [Reference on page 1438](#) in this topic for the full reference for available hooks and methods available to them.

- [Configuring hooks on page 1434](#)
- [Defining hooks for a resource on page 1435](#)
- [Before hooks on page 1436](#)
- [After hooks on page 1437](#)
- [Error handling on page 1437](#)
- [Defining common hooks on page 1437](#)
- [Reference on page 1438](#)

## Configuring hooks

The directory of the `hooks` folder may be changed by providing a value for `mws.hooks.location` in the configuration file. If the directory starts with a path separator (ie `/path/to/hooks`), it will be treated as an absolute path. Otherwise, it will be used relative to the location of the MWS home directory (for more information, see [Configuring Moab Web Services on page 1395](#)).

For example, if the MWS home directory is set to `/opt/mws`, the hooks directory by default would be in `/opt/mws/hooks`. Changing the `mws.hooks.location` property to `myhooks` would result in the hooks directory being located at `/opt/mws/myhooks`. Due to the default location of the MWS home directory, the default directory of the hooks directory is `/opt/mws/hooks`.

On startup, if the `hooks` directory does not exist, it will be created with a simple `README.txt` file with instructions on how to create hooks, the objects available, and the hooks available. If the folder or file is unable to be created, a message will be printed on the log with the full location of a `README` file, copied into a temporary directory.

## Defining hooks for a resource

Hooks are defined for resources by creating groovy class files in the hooks directory (`MWS_HOME/hooks` by default). Each groovy file must be named by the resource URL it is associated with and end in ".groovy". The following table shows some possible hook files that may be created. Notice that the virtual machines hook file is abbreviated as `vms`, just as the URL for virtual machines is `/rest/vms`. In most cases, the hook file names will exactly match the URLs. However, in cases of nested URLs—such as with "accounting/users"—the hook file name must replace slashes with periods. For example:

Resource	Hook filename
<b>Jobs</b>	<code>jobs.groovy</code>
<b>Nodes</b>	<code>nodes.groovy</code>
<b>Virtual Machines</b>	<code>vms.groovy</code>
<b>Accounting Users</b>	<code>accounting.users.groovy</code>
<b>Accounting Funds Reports Statement</b>	<code>accounting.funds.reports.statement.groovy</code>
<b>Accounting Charge Rates</b>	<code>accounting.charge-rates.groovy</code>
<i>url</i>	<i>url.groovy</i>

**i** `plugins.rm.groovy` is a valid hook filename. It works for the following URL:  
`/rest/plugins/<pluginID or all>/rm/<query or action>` (for example,  
`/rest/plugins/plugin1/rm/cluster-query`).

A complete example of a hook file is as follows:

## Complete Hook File

```

// Example before hook
def beforeList = {
  // Perform actions here
  // Return true to allow the API call to execute normally
  return true
}

def beforeShow = {
  // Perform actions here
  // Render messages to the user with a 405 Method Not Allowed
  // HTTP response code
  renderMessages("Custom message here", 405)
  // Return false to stop normal execution of the API call
  return false
}

// Example after hook
def afterList = { o ->
  if (!isSuccess()) {
    // Handle error here
    return false
  }
  // Perform actions here
  return o
}

```

**i** You must convert all actions or queries that are separated by dashes to a camel case. For example, the hooks called for "cluster-query" should be `beforeClusterQuery` and `afterClusterQuery`.

As the specific format for the hooks for `before` and `after` are different, each will be explained separately.

## Before hooks

As shown above, before hooks require no arguments. They can directly act on several properties, objects, and methods as described in [Reference on page 1438](#). The return value is one of the most important aspects of a before hook. If it is false, a `renderMessages`, `renderObject`, `renderList`, `render`, or `redirect` method *must* first be called. This signifies that the API call should be interrupted and the render or redirect action specified within the hook is to be completed immediately.

A return value of `true` signifies that the API call should continue normally. Parameters, session variables, request and response variables may all be modified within a `before` hook.

**i** If no return value is explicitly given, the result of the last statement in the `before` hook to be executed will be returned. This may cause unexpected behavior if the last statement resolves to `false`.

For all methods available to `before` hooks as well as specific examples, see [beforeSave on page 1438](#).

## After hooks

`After` hooks are always passed one argument: the object or list that is to be rendered as JSON. This may be modified as desired, but note that the object or list value is either a [JSONArray](#) or [JSONObject](#). Therefore, it may not be accessed and modified as a typical groovy Map.

Unlike `before` hooks, `after` hooks should not call the `render*` methods directly. This method will automatically be called on the resulting object or list returned. The `redirect` and `render` methods should also not be called at this point. Instead, if a custom object or list is desired to be used, the `serializeObject` and `serializeList` methods are available to create suitable results to return.

The return value of an `after` hook may be one of two possibilities:

- The potentially modified object or list passed as the first argument to the hook. In this case, this value will override the output object or list unless it is null.
- Null or false. In this case, the original, unmodified object or list will be used in the output.



The return value of the `after` hook, if not null or false, *must* be the modified object passed into the hook or an object or list created with the `serialize*` methods.

For all methods available to `after` hooks as well as specific examples, see [afterSave on page 1438](#).

## Error handling

`After` hooks, unlike the `before` hooks, have the possibility of handling errors encountered during the course of the request. Handling errors is as simple as adding a one-line check to the hook as shown above or in the following code:

```
if (!isSuccess()) {
    // Handle error
    return false
}
```

It is recommended that each `after` hook contain at least these lines of code to prevent confusion on what the input object or list represents or should look like.

The `isSuccess()` function is false if and only if the HTTP response code is 400 or higher, such as a 404 Not Found, 400 Bad Request, or 500 Internal Server Error and the cause of the error state was not in the associated `before` hook. In other words, objects and lists rendered in the `before` hook with any HTTP response code will never run the associated `after` hook.

When handling errors, the passed in object will always contain a `messages` property containing a list of strings describing the error(s) encountered.

## Defining common hooks

Sometimes it is beneficial to create hooks which are executed for all calls of a certain type, such as a `beforeList` hook that is executed during the course of listing any resource. These are possible using an `all.groovy` file. The format of this file is exactly the same as other hook files. The order of execution is as follows:

1. Before common hook executed.
2. Before resource-specific hook executed.
3. Normal API call executed.
4. After resource-specific hook executed.
5. After common hook executed.

## Reference

This page gives specific examples and reference for implementing hooks in MWS.

### [Available hooks](#)

The following table lists the available hooks for each resource with their associated HTTP method and description.

Name	HTTP method	Description
<b>beforeList</b>	GET	Runs before an API call that lists resources (for example, GET /rest/jobs).
<b>afterList</b>	GET	Runs after an API call that lists resources.
<b>beforeShow</b>	GET	Runs before an API call that returns a single resource (for example, GET /rest/jobs/job.1).
<b>afterShow</b>	GET	Runs after an API call that returns a single resource.
<b>beforeSave</b>	POST	Runs before an API call that saves a new resource (for example, POST /rest/-jobs).
<b>afterSave</b>	POST	Runs after an API call that returns a single resource.
<b>beforeUpdate</b>	PUT	Runs before an API call that returns a single resource (for example, PUT /rest/jobs/job.1).
<b>afterUpdate</b>	PUT	Runs after an API call that returns a single resource.
<b>beforeDelete</b>	DELETE	Runs before an API call that returns a single resource (for example, DELETE /rest/jobs/job.1).
<b>afterDelete</b>	DELETE	Runs after an API call that returns a single resource.

**i** If a resource does not support a certain operation, any hooks for that operation will simply be ignored—such as [beforeSave](#) and [afterSave](#) hooks for the Node resource, where saving is not supported.

### Available properties

The following table lists the properties, objects, and methods available in all hooks. Note that although it is possible to directly call the `render*` methods in the `after` hooks, it is not recommended.

Name	Type	Description
<b>params</b>	Map	Contains all URL parameters as well as the body of the request as parsed JSON.
<b>request</b>	<a href="#">HttpServletRequest</a>	Contains properties of the HTTP request.
<b>response</b>	<a href="#">HttpServletResponse</a>	Contains properties of the HTTP response which can be modified directly.
<b>session</b>	<a href="#">HttpSession</a>	Contains the session parameters which can be modified directly.
<b>flash</b>	Map	Temporary storage that stores objects within the session for the next request only.
<b>controllerName</b>	String	The name of the controller responding to the request. Only available in <code>before</code> hooks.
<b>actionName</b>	String	The name of the action to be run on the controller. Only available in <code>before</code> hooks.
<b>apiVersion</b>	String	The API version for the current request (for example, 1 for 7.0 and 7.1, 2 for 7.2).

**i** The parsed JSON may be accessed in `before` hooks as a simple groovy Map with `params[controllerName]`.

In addition, several methods are available to the hooks. These are described in the following sections.

### Redirect

The `redirect` method may be used to redirect the request to another API call or an arbitrary URL.

```
redirect(uri:'/rest/jobs')           // uri is used for internal redirection within MWS
redirect(url:'http://adaptivecomputing.com') // url is used for external redirection
redirect(uri:'http://adaptivecomputing.com', params:[lang:'en']) // params may be used
for URL parameters
```

**i** The `redirect` method will use the GET HTTP method for the resulting redirected request.

See the `redirect` method's [documentation](#) for more information.

### Rendering objects, lists, or messages

There are several `render*` methods available to handle any case where objects or lists are desired to be rendered directly from the hook without continuing to the API call. Three different methods may be used depending on the desired output object type:

#### Render object

---

```
// Object that should be rendered as JSON
def objectToRender = ...
// HTTP response code (bad request)
def responseCode = 400
// Render a simple object
renderObject(objectToRender)
// Render a simple object with a custom response code
renderObject(objectToRender, responseCode)
```

#### Render list

---

```
// List that should be rendered as JSON
def listToRender = ...
// If the totalCount property differs from resultCount, use this value instead
def totalCount = ...
// HTTP response code (bad request)
def responseCode = 400
// Render a simple list
//   Dynamically adds "resultCount" and "totalCount" properties based on the size of
//   the input list
renderList(listToRender)
// Render a simple list with a custom "totalCount"
renderList(listToRender, totalCount)
// Render a simple list without changing the "totalCount" but with a custom response
//   code
renderList(listToRender, null, responseCode)
// Render a simple list with a custom "totalCount" and response code
renderList(listToRender, totalCount, responseCode)
```

#### Render message(s)

---

```
// Messages
def messageToRender = "Single message"
def messagesListToRender = ["Message 1", "Message 2"]
// HTTP response code (bad request)
def responseCode = 400
// Render messages as an object with a property of "messages" containing a list of the
//   messages passed in
renderMessages(messageToRender)
renderMessages(messageToRender, responseCode)
// Supports either a single String or list of Strings
renderMessages(messagesListToRender)
renderMessages(messagesListToRender, responseCode)
```

**i** It is not recommended to call any of these methods from an `after` hook.

### Render

Less commonly used, the `render` method is also available directly. This may be used to render text directly, change the content-type of the output, and many other functions. See the `render` method's [documentation](#) for more information.

**i** It is not recommended to call this method from an `after` hook.

### Serialize objects

The `serializeObject` and `serializeList` methods may be used to convert a custom object or list respectively into a format usable for returning in the `after` hooks. Simply pass in the object or list and a serialized version will be returned from the method.

```
def afterShow = {
    def objectToRender = ...
    def serializedObject = serializeObject(objectToRender)
    return serializedObject
}
```

```
def afterShow = {
    def listToRender = [...]
    def serializedList = serializeList(listToRender)
    return serializedList
}
```

### Error handling

Error handling is only available in `after` hooks by using the following check:

```
if (!isSuccess()) {
    // Handle error
    return ... // False or modified object/list to render
}
```

### Usage examples

#### **Override an API call**

The following hook would serve to override an entire API call, the list call in this case, and return a `messages` list containing a single element of "Action is not supported" and a HTTP response code of 405 (Method Not Allowed):

```
def beforeList = {
    renderMessages("Action is not supported", 405)
    return false
}
```

To be even more specific and disallow the deletion of virtual machines, the following may be used as the `vms.groovy` file:

```
def beforeDelete = {
    renderMessages("Virtual Machine deletion is not allowed", 405)
    return false
}
```

### Add an additional property during job creation

To add an additional property to a job definition during creation, create a `beforeSave` hook in the `jobs.groovy` file as follows:

```
def beforeSave = {
    // params[controllerName] is equivalent to params["job"] or params.job
    params[controllerName].user = "myuser"
}
```

This would cause the created job to have a user of `myuser`.

### Redirect based on URL parameter

To redirect an API call if a certain URL parameter exists, create a `beforeSave` hook in the `jobs.groovy` file as follows:

```
def beforeSave = {
    if (params.external) {
        redirect(url:'http://example.com/create-job')
        return false; // Stop API call
    }
}
```

This would cause an API call of `PUT /rest/jobs?external=1` to redirect to `GET http://example.com/create-job`.

### Remove a property from getting a single job

To remove a property from the output of getting a single job, create an `afterShow` hook in the `jobs.groovy` file as follows:

```
def afterShow = { o ->
    o.discard("group")
    return o
}
```

This will cause the resulting JSON to be missing the `group` property of the job resource. Note again that these calls must use the [JSONArray](#) and [JSONObject](#) classes as mentioned in [After hooks on page 1437](#).

### Filter list items

To filter the items in a list nodes request based on user provided query parameter in the URL, use the following in the `nodes.groovy` file. A sample request that would activate the filter is `http://localhost:8080/mws/rest/nodes?api-version=3&filter-power=On`.

```

def afterList = { o ->
    // Do not filter if the user did not ask for it
    if (!params['filter-power'])
        return o
    // o = {resultCount: x, totalCount: x, results:[...]}

    // Using a built-in groovy method findAll to return all
    // list items that return true from the block
    def results = o.results.findAll { node ->
        // Includes the node only if the power equals the user input
        return params['filter-power'].equalsIgnoreCase(node.power)
    }

    // Sets the results on the return object and updates the counts
    o.element("results", results)
    o.element("resultCount", results.size())
    return o
}

```

To filter the items in a list nodes request based on values within the list itself, such as variable values, use the following in the `nodes.groovy` file.

```

def afterList = { o ->
    // o = {resultCount: x, totalCount: x, results:[...]}
    // Using a built-in groovy method findAll to return all
    // list items that return true from the block
    def results = o.results.findAll { node ->
        // Includes the node only if the variable "included" is set to "true"
        return node.variables?.included=="true"
    }

    // Sets the results on the return object and updates the counts
    o.element("results", results)
    o.element("resultCount", results.size())
    return o
}

```

### Related topics

- [About the API on page 1423](#)

## Authentication

MWS uses Basic Authentication for all REST API requests. This means that a username and password must be provided for each call to resources. There are two types of accounts that can be granted access: **Users** and **Applications**.

- For instructions on how to set the credentials for the default **User** account, see [Setting up MWS security on page 1410](#).
- For instructions on how to manage **Application** accounts, see [Access control on page 1420](#).

To use Basic Authentication, each client request must contain a header that looks like this:

```
Authorization: Basic YWRhcHRpdmU6YzNVU3R1bkU=
```

The string after the word `Basic` is the base64 encoding of `username : password`. In the example above, `YWRhcHRpdmU6YzNVU3R1bkU=` is the base64 encoding of `adaptive:c3UStunE`. For more details, see section 2 of [RFC 2617](#).

 The username and password in the Basic Authentication header are encoded but not encrypted. Therefore, it is *strongly* recommended that MWS be run behind a proxy (like Apache) with SSL enabled. For more information, see [Setting up MWS security on page 1410](#).

#### Related topics

- [About the API on page 1423](#)

## System events

The broad category of system events may be broken down into two subcategories: events and notification conditions.

- [Events on page 1444](#)
- [Notification conditions on page 1445](#)

### Events

[Events on page 1528](#) are created by many components in the system, but most events originate from Moab Workload Manager and Moab Web Services. Events can be [created via the MWS interface](#) or by being placed on the message queue. The ZeroMQ™ message queue libraries were introduced in Moab and MWS 7.5.0. The message queue is critical to service lifecycle functionality (see "Service lifecycle" in the *Moab Cloud Suite Installation Guide*).

In a typical system, Moab will communicate events to MWS via a "private" message queue, and then MWS will replicate the events on the "public" message queue, or the message queue that is available to subscribers with the correct secret keys. In some cases, such as those related to the MWS service lifecycle, MWS uses events to determine activities or capabilities that are available.

A typical message on the message queue may look like the following (sent with a topic of `system.moab`):

Sample message on message queue

```
{
  "body" : {
    "associatedObjects" : [
      {
        "id" : "Moab",
        "type" : "scheduler"
      }
    ],
    "code" : 16777619,
    "eventDate" : "2014-02-28T10:57:21.000-0700",
    "message" : "A scheduler iteration is ending.",
    "origin" : "MSysMainLoop.c, MSysMainLoop, line 959"
  },
  "messageId" : "843269550",
  "messageType" : "event",
  "senderId" : "mwm@mwm-server",
  "sentDate" : "2014-02-28T10:57:21.000-0700",
  "ttl" : 3000
}
```

## Notification conditions

[Notification conditions on page 1580](#) are related to an event, but differ in three distinct areas:

1. Notification conditions are a persistent condition of the system or a component rather than a single occurrence.
  - They are ongoing rather than reoccurring, which is why they are generated from NotificationConditions.
  - They may be observed many times, but the condition is always the same.
  - A good test for this is if something "is" wrong rather than something "went" wrong.
2. Notification conditions can be acted on to result in a resolved state, mean the administrator or user can and must take actions to "fix" the condition or problem.
3. Notification conditions contain state information based on administrator or user input, meaning that they contain information about the condition (similar to events), but also contain the "status" of the administrator's view of the notification, whether it is currently open, dismissed, or ignored.

In general, questions may be asked to ascertain whether an event or a notification condition is the right fit for an occurrence. These questions, along with some sample situations, are provided below.

- Is the occurrence the root cause of a potentially ongoing condition?
  - A VM migration failed because the VM's state was unknown. The root cause was that the state was unknown, not that the VM migration failed. Therefore, VM migration failed would be an event, while the unknown state would be a notification condition.
  - A VM service provision fails because there are no hypervisors that satisfy the requirements. This would be an event. Note that there may be a notification related to this failure, such as a service template requires a feature that does not exist on any

hypervisors in the system, but this would be distinctly detected and managed from the provision failure event.

- A request to MWS failed because the connection between Moab and MongoDB was misconfigured. The failed request may be represented as an event, but a notification condition should exist that the connection between Moab and MongoDB was down.
- Can an administrator or user affect the outcome of the occurrence?
  - The outcome of a VM migration failing in the past and cannot be changed by the administrator. However, the outcome of a future VM migration may be changed when the administrator resolves the root problem (such as VM state is unknown).

#### Related topics

- [Events on page 1528](#)
- [Notifications on page 1585](#)
- [Notification conditions on page 1580](#)
- [Securing the connection with the message queue on page 1417](#)
- [Creating events and notifications on page 1704](#) (for plugin development only)
- [Plugin event service on page 1755](#)

## Resources

### Resources introduction

The sections in this chapter show the MWS resources and the HTTP methods defined on them. The prefix for these resources depends on how the `mws.war` file is deployed. A typical prefix would be `http://localhost:8080/mws`. Using this example, one absolute resource URI would be `http://localhost:8080/mws/rest/jobs`.



This section only contains documentation for the latest API version. Please see the table in [Requesting specific API versions on page 1428](#) for links to documentation for previous versions.

This chapter contains these sections:

- [Access control lists \(ACLs\) on page 1448](#)
- [Accounting Accounts on page 1451](#)
- [Accounting Allocations on page 1455](#)
- [Accounting Charge rates on page 1459](#)
- [Accounting Funds on page 1463](#)
- [Accounting Liens on page 1473](#)
- [Accounting Organizations on page 1477](#)

- [Accounting Quotes](#) on page 1480
- [Accounting Transactions](#) on page 1483
- [Accounting Usage records](#) on page 1488
- [Accounting Users](#) on page 1502
- [Credentials](#) on page 1506
- [Diagnostics](#) on page 1521
- [Distinct](#) on page 1526
- [Events](#) on page 1528
- [Images](#) on page 1536
- [Job arrays](#) on page 1545
- [Jobs](#) on page 1547
- [Job templates](#) on page 1569
- [Metric types](#) on page 1571
- [Nodes](#) on page 1573
- [Notifications](#) on page 1585
- [Notification conditions](#) on page 1580
- [Permissions](#) on page 1593
- [Plugins](#) on page 1599
- [Plugin types](#) on page 1607
- [Policies](#) on page 1611
- [Principals](#) on page 1627
- [Priority](#) on page 1633
- [Reports](#) on page 1636
- [Reservations](#) on page 1646
- [Resource types](#) on page 1654
- [Roles](#) on page 1655
- [Standing reservations](#) on page 1661

#### Related topics

- [Resources reference](#) on page 1782

# Access control lists (ACLs)

This topic describes behavior of the ACL Rules (Access Control List Rules) object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Access Control Lists \(ACLs\)](#) reference contains the type and description of all fields in the **ACL Rules** object. It also contains details regarding which fields are valid during PUT and POST actions.

## Supported methods

**i** ACLs are not directly manipulated through a single URL, but with sub-URLs of the other objects such as Virtual Containers and Reservations.

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/reservations/&lt;rsvid&gt;/acl-rules/&lt;aclId&gt;</a>	--	<a href="#">Create or update ACL</a>	--	<a href="#">Delete ACL</a>
<a href="#">/rest/vcs/&lt;vcId&gt;/acl-rules/&lt;aclId&gt;</a>	--	<a href="#">Create or update ACL</a>	--	<a href="#">Delete ACL</a>

This topic contains these sections:

- [Getting ACLs on page 1448](#)
- [Creating or updating ACLs on page 1449](#)
  - [Create or update ACL on page 1449](#)
- [Deleting ACLs on page 1450](#)
  - [Delete ACL on page 1450](#)

## Getting ACLs

Although **ACL Rules** cannot be retrieved directly using the GET method on any of the `acl-rules` resources, **ACL Rules** are attached to supported objects when querying for them. Each supported object contains a field named `aclRules`, which is a collection of the **ACL Rules** defined on that object.

## Supported objects

The following is a list of objects that will return **ACL Rules** when queried:

- [Reservations on page 1646](#)
- [Standing reservations on page 1661](#)

## Creating or updating ACLs

The HTTP PUT method is used to create or update **ACL Rules**. The request body can contain one or more **ACL Rules**. If an **ACL Rule** with the same `type` and `value` exists, then it will be overwritten.

### Quick reference

```
PUT http://localhost:8080/mws/rest/reservations/<rsvId>/acl-rules?api-version=3
```

## Create or update ACL

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/reservations/<rsvId>/acl-rules?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>objectId</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

The request body below shows all the fields that are available for the PUT method, along with some sample values.

JSON Request Body

```
{
  "aclRules": [
    {
      "affinity": "POSITIVE",
      "comparator": "LEXIGRAPHIC_EQUAL",
      "type": "USER",
      "value": "ted"
    }
  ]
}
```

### Sample response



This message may not match the message returned from Moab exactly, but is given as an example of the structure of the response.

JSON Request Body

```
{"messages":["Reservation 'rsv1' successfully modified"]}
```

### Samples

Create or update multiple ACLs on a single object:

```

CPUT http://localhost:8080/mws/rest/reservations/system.21/acl-rules?api-version=3
-----
{"aclRules": [
  {
    "affinity": "POSITIVE",
    "comparator": "LESS_THAN_OR_EQUAL",
    "type": "DURATION",
    "value": "3600"
  },
  {
    "affinity": "POSITIVE",
    "comparator": "LEXIGRAPHIC_EQUAL",
    "type": "USER",
    "value": "ted"
  }
]}

```

Restrictions

- **ACL Rules** cannot be added to or updated on **Standing Reservations**.

## Deleting ACLs

The HTTP DELETE method is used to remove **ACL Rules**.

Quick reference

**i** **ACL Rules** cannot be removed from **Standing Reservations**.

```

DELETE http://localhost:8080/mws/rest/reservations/<rsvId>/acl-rules?api-
version=3/<aclId>

```

## Delete ACL

URLs and parameters

```

DELETE http://localhost:8080/mws/rest/reservations/<objectId>/acl-rules?api-
version=3/<aclId>

```

Parameter	Required	Type	Valid values	Description
<b>objectId</b>	Yes	String	--	The unique identifier of the object from which to remove the <b>ACL Rule</b> .
<b>aclId</b>	Yes	String	--	A string representing the <b>ACL Rule</b> , with the format type:value.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

**i** This message may not match the message returned from Moab exactly, but is given as an example of the structure of the response.

```
JSON Response
-----
{"messages":["Successfully modified reservation 'rsv1'"]}
```

Restrictions

- **ACL Rules** cannot be removed from **Standing Reservations**.

Related topics

- [Fields: Access Control Lists \(ACLs\) on page 1783](#)
- [Resources introduction on page 1446](#)

# Accounting

## Accounting Accounts

This section describes the services available through Moab Web Services for interacting with the **Account** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Accounts](#) reference contains the type and description of the default fields for the **Accounts** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/accounts</a>	<a href="#">Get all accounts</a>	--	--	--
<a href="#">/rest/accounting/accounts/&lt;id&gt;</a>	<a href="#">Get single account</a>	--	--	--

This topic contains these sections:

- [Getting accounts on page 1452](#)
  - [Get all accounts on page 1452](#)
  - [Get single account on page 1454](#)

## Getting accounts

The HTTP GET method is used to retrieve **Accounts** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/accounts?api-version=3
GET http://localhost:8080/mws/rest/accounting/accounts/<id>?api-version=3
```

## Get all accounts

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/accounts?api-version=3&proxy-user=<user>
[&query=<query_conditions>][&fields=<fields_to_display>][&sort=<fields_to_sort>]|&show-
all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>query</b>	No	JSON	--	Results are restricted to those having the specified field values.	query={"organization":"sciences"}

**i** The **query** parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.

Parameter	Required	Type	Valid values	Description	Example
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,organization
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	sort={"organization":1}
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/accounts?api-version=3&proxy-user=amy&fields=id,organization&pretty=true
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "organization": "sciences",
      "id": "biology"
    },
    {
      "organization": "sciences",
      "id": "chemistry"
    }
  ]
}
```

## Get single account

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/accounts/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	--
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,organization
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/accounts/chemistry?api-version=3&proxy-user=amy&pretty=true
-----
{
  "id": "chemistry",
  "active": true,
  "organization": "",
  "description": "Chemistry Dept",
  "users": [
    {
      "id": "amy",
      "active": true,
      "admin": false
    },
    {
      "id": "bob",
      "active": true,
      "admin": false
    },
    {
      "id": "dave",
      "active": true,
      "admin": false
    }
  ]
}
```

Related topics

- [Fields: Accounts on page 1796](#)
- [Resources introduction on page 1446](#)

## Accounting Allocations

This section describes the services available through Moab Web Services for interacting with the **Allocation** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Allocations](#) reference contains the type and description of the default fields for the **Allocation** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/accounting/allocations</code>	<a href="#">Get all allocations</a>	--	--	--
<code>/rest/accounting/allocations/&lt;id&gt;</code>	<a href="#">Get single allocation</a>	--	--	--

This topic contains these sections:

- [Getting allocations on page 1456](#)
  - [Get all allocations on page 1456](#)
  - [Get single allocation on page 1458](#)

## Getting allocations

The HTTP GET method is used to retrieve **Allocation** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/allocations?api-version=3
GET http://localhost:8080/mws/rest/accounting/allocations/<id>?api-version=3
```

### Get all allocations

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/allocations?api-version=3&proxy-user=<user>[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>query</b>	No	JSON	--	Results are restricted to those having the specified field values. <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.                 </div>	query={"active":true}
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,fund,amount

Parameter	Required	Type	Valid values	Description	Example
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	<code>sort={"fund":1}</code>
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	<code>show-all=true</code>

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/accounting/allocations?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "totalCount": 5,
  "resultCount": 5,
  "results": [
    {
      "id": 1,
      "fund": 1,
      "startTime": "2013-07-12 22:16:33 UTC",
      "endTime": "infinity",
      "amount": 50000000,
      "creditLimit": 0,
      "initialDeposit": 50000000,
      "allocated": 50000000,
      "active": true,
      "description": ""
    },
    {
      "id": 3,
      "fund": 3,
      "startTime": "2013-07-12 22:16:33 UTC",
      "endTime": "infinity",
      "amount": 0,
      "creditLimit": 20000000,
      "initialDeposit": 0,
      "allocated": 0,
      "active": true,
      "description": ""
    },
    {
      "id": 2,
      "fund": 2,
      "startTime": "2013-07-12 22:16:33 UTC",
      "endTime": "infinity",
      "amount": 30000000,
      "creditLimit": 0,
      "initialDeposit": 30000000,
      "allocated": 30000000,
      "active": true,
      "description": ""
    }
  ]
}
```

### Get single allocation

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/allocations/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	--
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id, fund, amount
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/accounting/allocations/1?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "id": 1,
  "fund": 1,
  "startTime": "2013-07-12 22:16:33 UTC",
  "endTime": "infinity",
  "amount": 50000000,
  "creditLimit": 0,
  "initialDeposit": 50000000,
  "allocated": 50000000,
  "active": true,
}
```

#### Related topics

- [Fields: Allocations on page 1800](#)
- [Resources introduction on page 1446](#)

## Accounting Charge rates

This section describes the services available through Moab Web Services for interacting with the **ChargeRate** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.



The [Fields: Charge Rates](#) reference contains the type and description of the default fields for the **ChargeRates** object.

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/charge-rates</a>	<a href="#">Get all charge rates</a>	--	--	--
<a href="#">/rest/accounting/charge-rates/&lt;name&gt;/&lt;value&gt;</a>	<a href="#">Get single charge rate</a>	--	--	--
<a href="#">/rest/accounting/charge-rates/&lt;name&gt;</a>	<a href="#">Get single charge rate</a>	--	--	--

This topic contains these sections:

- [Getting charge rates on page 1460](#)
  - [Get all charge rates on page 1460](#)
  - [Get single charge rate on page 1462](#)

## Getting charge rates

The HTTP GET method is used to retrieve **ChargeRate** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/charge-rates?api-version=3
GET http://localhost:8080/mws/rest/accounting/charge-rates?api-version=3/<name>
[/<value>]
```

## Get all charge rates

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/charge-rates?api-version=3&proxy-
user=<user>[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_
sort>]|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy

Parameter	Required	Type	Valid values	Description	Example
<b>query</b>	No	JSON	--	<p>Results are restricted to those having the specified field values.</p> <div style="border: 1px solid #0056b3; padding: 5px; margin-top: 10px;"> <p><b>i</b> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	<code>query={"name":"QualityOfService"}</code>
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	<code>fields=id,organization</code>
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	<code>sort={"organization":1}</code>

Parameter	Required	Type	Valid values	Description	Example
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/accounting/charge-rates?api-version=3&proxy-user=moab&pretty=true
```

```
-----
{
  "totalCount": 4,
  "resultCount": 4,
  "results": [
    {
      "name": "Processors",
      "value": "",
      "amount": "1/s",
      "description": "1 credit per processor-second"
    },
    {
      "name": "QualityOfService",
      "value": "high",
      "amount": "*2",
      "description": "Charge double for high QOS"
    },
    {
      "name": "QualityOfService",
      "value": "low",
      "amount": "*.5",
      "description": "Charge half for low QOS"
    },
    {
      "name": "QualityOfService",
      "value": "",
      "amount": "*1",
      "description": "No extra charge for \"normal\" QOSes"
    }
  ]
}
```

### Get single charge rate

**i** A regular charge rate is uniquely specified by both its name and its value. A default charge rate has a null value and is uniquely specified by only its name.

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/charge-rates?api-version=3/<name>
[/<value>]?proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>name</b>	Yes	String	--	The name of the charge rate.	--
<b>value</b>	No	String	--	The value of the charge rate.	--
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=name,value,amount
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/charge-rates/QualityOfService/high?api-version=3&proxy-user=moab&pretty=true
-----
{
  "name": "QualityOfService",
  "value": "high",
  "amount": "*2",
  "description": "Charge double for high QOS"
}
```

Related topics

- [Fields: Charge Rates on page 1806](#)
- [Resources introduction on page 1446](#)

## Accounting Funds

This section describes the services available through Moab Web Services for interacting with the **Fund** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Funds](#), [Fields: Fund Balances](#), [Fields: Fund Statements](#), and [Fields: Fund Statement Summary](#) reference sections contain the type and description of the default fields in the **Fund** object as well as related objects and reports given in the URLs below.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/funds</a>	<a href="#">Get all funds</a>	--	--	--
<a href="#">/rest/accounting/funds/&lt;id&gt;</a>	<a href="#">Get single fund</a>	--	--	--
<a href="#">/rest/accounting/funds/balances</a>	<a href="#">Get all fund balances</a>	--	--	--
<a href="#">/rest/accounting/funds/reports/statement</a>	<a href="#">Get fund statement</a>	--	--	--
<a href="#">/rest/accounting/funds/reports/statement/summary</a>	<a href="#">Get fund statement summary</a>	--	--	--

This topic contains these sections:

- [Getting funds on page 1464](#)
  - [Get all funds on page 1465](#)
  - [Get single fund on page 1467](#)
  - [Get all fund balances on page 1469](#)
  - [Get fund statement on page 1471](#)
  - [Get fund statement summary on page 1472](#)

## Getting funds

The HTTP GET method is used to retrieve **Fund** information.

Quick reference

```
GET http://localhost:8080/mws/rest/accounting/funds?api-version=3
GET http://localhost:8080/mws/rest/accounting/funds/<id>?api-version=3
GET http://localhost:8080/mws/rest/accounting/funds/balances?api-version=3
GET http://localhost:8080/mws/rest/accounting/funds/reports/statement?api-version=3
GET http://localhost:8080/mws/rest/accounting/funds/reports/statement/summary?api-version=3
```

## Get all funds

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/funds?api-version=3&proxy-user=<user>
[&active=true][&filter=<filter_options>[&filter-type=<filter_type>]][&query=<query_
conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]|&show-all=
(true|false)]
```

Parameter	Required	Type	Description	Example
<b>proxy-user</b>	Yes	String	Perform action as defined MAM user.	proxy-user=amy
<b>active</b>	No	Boolean	Lists only active or non-active allocations of the fund. The fund amount becomes the sum of the active/inactive allocations.	active=true
<b>filter</b>	No	JSON	Query funds based on defined MAM filter.	filter={"account":"chemistry"}
<b>filter-type</b>	No	String	Query funds based on defined MAM filter type.	filter-type=NonExclusive

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	<p>Results are restricted to those having the specified field values.</p> <div style="border: 1px solid #004a7c; padding: 5px; margin-top: 10px;"> <p> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	<pre>query={"priority":"2","allocation.active":"false"}</pre>
<b>fields</b>	No	String	Comma-separated list of field names to display.	<pre>fields=id,name,amount</pre>
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	<pre>sort={"id":1}</pre>

Parameter	Required	Type	Description	Example
<b>show-all</b>	No	Boolean (true or false)	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/funds?api-version=3&proxy-user=amy&fields=id,name,amount&pretty=true
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "id": 1,
      "name": "biology",
      "amount": 50000000
    },
    {
      "id": 2,
      "name": "chemistry",
      "amount": 99727
    }
  ]
}
```

**Get single fund**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/funds/<id>?api-version=3&proxy-user=<user>[&active=(true|false)] [&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Description	Example
<b>id</b>	Yes	String	The unique identifier of the object	--
<b>proxy-user</b>	Yes	String	Perform action as defined MAM user.	proxy-user=amy

Parameter	Required	Type	Description	Example
<b>active</b>	No	Boolean	Lists only active or non-active allocations of the fund. The fund amount becomes the sum of the active/inactive allocations.	active=true
<b>fields</b>	No	String	Comma-separated list of field names to display.	fields=id,name,amount
<b>show-all</b>	No	Boolean (true or false)	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/accounting/funds/1?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "id": 1,
  "name": "biology",
  "priority": 0,
  "defaultDeposit": 50000000,
  "description": "",
  "amount": 50000000,
  "allocated": 50000000,
  "initialDeposit": 50000000,
  "creditLimit": 0,
  "allocations": [
    {
      "id": 1,
      "startTime": "2013-08-21 16:57:53 UTC",
      "endTime": "infinity",
      "amount": 50000000,
      "creditLimit": 0,
      "initialDeposit": 50000000,
      "allocated": 50000000,
      "active": false,
      "description": ""
    }
  ],
  "fundConstraints": [ {
    "id": 1,
    "name": "Account",
    "value": "biology"
  } ]
}
```

## Get all fund balances

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/funds/balances?api-version=3&proxy-user=<user>[&filter=<filter_options>][&filter-type=<filter_type>]
```

Parameter	Required	Type	Description	Example
<b>proxy-user</b>	Yes	String	Perform action as defined MAM user.	proxy-user=amy
<b>filter</b>	No	JSON	Query funds based on defined MAM filter.	filter={"account": "chemistry"}
<b>filter-type</b>	No	String	Query funds based on defined MAM filter type.	filter-type-e=NonExclusive

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

The fund balances resource is an aggregation of fund data. For more information, see the [Fields: Fund Balances on page 1809](#) reference section.

```
GET http://localhost:8080/mws/rest/accounting/funds/balances?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "id": 2,
      "name": 1204,
      "priority": 0,
      "description": "R&D for Manufacturing",
      "creationTime": "2012-02-02 09:34:42 UTC",
      "amount": 9060000,
      "deposited": 9060000,
      "creditLimit": 0,
      "reserved": 0,
      "allocations": [
        {
          "id": 2,
          "amount": 9060000,
          "creditLimit": 0,
          "deposited": 9060000
        }
      ],
      "fundConstraints": [
        {
          "id": 2,
          "name": "CostCenter",
          "value": 1204
        }
      ],
      "balance": 9060000,
      "available": 9060000,
      "allocated": 9060000,
      "used": 0,
      "percentRemaining": 100,
      "percentUsed": 0
    },
    {
      "id": 5,
      "name": "",
      "priority": 0,
      "description": "",
      "creationTime": "2012-04-03 09:25:47 UTC",
      "amount": 901290219001,
      "deposited": 901290219021,
      "creditLimit": 30,
      "reserved": 84018308897.68,
      "allocations": [
        {
          "id": 6,
          "amount": 901290219001,
          "creditLimit": 30,
          "deposited": 901290219021
        }
      ],
      "fundConstraints": [],
      "balance": 817271910103.32,
      "available": 817271910133.32,
      "allocated": 901290219051,

```

```

    "used": 20,
    "percentRemaining": 100,
    "percentUsed": 0
  }
]
}

```

## Get fund statement

### URLs and parameters

```

GET http://localhost:8080/mws/rest/accounting/funds/reports/statement?api-
version=3&proxy-user=<user>[&filter=<filter_options>][&filter-type=<filter_type>]
[&start-time=<date_string>][&end-time=<date_string>][&context=<context>]

```

Parameter	Required	Type	Description	Example
<b>proxy-user</b>	Yes	String	Perform action as defined MAM user.	proxy-user=amy
<b>filter</b>	No	JSON	Query funds based on defined MAM filter.	filter={"account": "chemistry"}
<b>filter-type</b>	No	String	Query funds based on defined MAM filter type.	filter-type=e=NonExclusive
<b>start-time</b>	No	Date, -infinity, or now	Filter allocations and transaction after a start time.	start-time=2012-04-03 15:24:39 UTC
<b>end-time</b>	No	Date, -infinity, or now	Filter allocations and transactions before an end time.	end-time=2012-04-03 15:24:39 UTC
<b>context</b>	No	hpc or cloud	The context to use in Moab Accounting Manager.	context=hpc

**i** The **context** parameter overrides the default context set for MAM using the `mam.context` configuration parameter. For more information about this parameter, see [Configuration on page 1772](#).

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

The fund statement report provides a snapshot of the current funds. For more information, see [Fields: Fund Statements on page 1835](#).

```
GET http://localhost:8080/mws/rest/accounting/funds/reports/statement?api-version=3&proxy-user=amy&fields=startBalance,endBalance&pretty=true

{
  "startBalance":1234.01,
  "endBalance":1000
}
```

**Get fund statement summary**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/funds/reports/statement/summary?api-version=3&proxy-user=<user>[&filter=<filter_options>] [&filter-type=<filter_type>] [&start-time=<date_string>] [&end-time=<date_string>]
```

Parameter	Required	Type	Description	Example
<b>proxy-user</b>	Yes	String	Perform action as defined MAM user.	proxy-user=amy
<b>filter</b>	No	JSON	Query funds based on defined MAM filter.	filter={"account":"chemistry"}
<b>filter-type</b>	No	String	Query funds based on defined MAM filter type.	filter-type-e=NonExclusive
<b>start-time</b>	No	Date, -infinity, or now	Filter allocations and transaction after a start time.	start-time=2012-04-03 15:24:39 UTC
<b>end-time</b>	No	Date, -infinity, or now	Filter allocations and transactions before an end time.	end-time=2012-04-03 15:24:39 UTC

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

The fund statement summary is slightly different from the typical fund statement in that the transactions are provided as summaries grouped by object and action. For more information, see [Fields: Fund Statement Summary on page 1819](#).

```
GET http://localhost:8080/mws/rest/accounting/funds/reports/statement/summary?api-
version=3&proxy-
user=amy&fields=totalCredits,totalDebits,transactions.action,transactions.amount,trans
actions.count&pretty=true
-----
{
  "totalCredits":200.02,
  "totalDebits":-100,
  "transactions":[ {
    "action":"Deposit",
    "amount":200.02,
    "count":2
  }, {
    "action":"Charge",
    "amount":-100,
    "count":1
  }
]
}
```

Related topics

- [Fields: Funds on page 1850](#)
- [Resources introduction on page 1446](#)

## Accounting Liens

This section describes the services available through Moab Web Services for interacting with the Lien object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

 The [Fields: Liens](#) reference contains the type and description of the default fields for the **Liens** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/liens</a>	<a href="#">Get all liens</a>	--	--	--
<a href="#">/rest/accounting/liens/&lt;id&gt;</a>	<a href="#">Get single lien</a>	--	--	--

This topic contains these sections:

- [Getting liens on page 1474](#)
  - [Get single lien on page 1476](#)
  - [Get all liens on page 1474](#)

## Getting liens

The HTTP GET method is used to retrieve **Lien** information.

### [Quick reference](#)

```
GET http://localhost:8080/mws/rest/accounting/liens?api-version=3
GET http://localhost:8080/mws/rest/accounting/liens/<id>?api-version=3
```

## Get all liens

### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/accounting/liens?api-version=3&proxy-user=<user>
[&active=true] [&filter=<filter_options> [&filter-type=<filter_type>]] [&query=<query_
conditions>] [&fields=<fields_to_display> [&sort=<fields_to_sort>] |&show-all=
(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>active</b>	No	Boolean	--	Lists only active or non-active liens.	active=true
<b>filter</b>	No	JSON	--	Query funds based on defined MAM filter.	filter={"account": "chemistry"}
<b>filter-type</b>	No	String	--	Query funds based on defined MAM filter type.	filter-type=NonExclusive

Parameter	Required	Type	Valid values	Description	Example
<b>query</b>	No	JSON	--	<p>Results are restricted to those having the specified field values.</p> <div style="border: 1px solid #0056b3; padding: 5px; margin-top: 10px;"> <p><b>i</b> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	<code>query={"allocations.fund":2}</code>
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	<code>fields=id,instance,amount</code>
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	<code>sort={"instance":1}</code>
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	<code>show-all=true</code>

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/liens?api-version=3&proxy-user=amy&filter={"account":"chemistry"}&fields=instance,amount&active=true&pretty=true
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "instance": "job.1",
      "amount": 57600
    },
    {
      "instance": "job.2",
      "amount": 40762
    }
  ]
}
```

**Get single lien**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/liens/<id>?api-version=3&proxy-user=<user>[&active=(true|false)][&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object	--
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>active</b>	No	Boolean	--	Lists only active or non-active liens.	active=true
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,name,amount
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/accounting/liens/1?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "id": 1,
  "instance": "job.1",
  "usageRecord": 1,
  "startTime": "2013-08-21 16:45:57 UTC",
  "endTime": "2013-08-21 17:45:57 UTC",
  "duration": 3600,
  "description": "",
  "amount": 57600,
  "allocations": [ {
    "id": 2,
    "fund": 2,
    "amount": 57600
  } ]
}
```

### Related topics

- [Fields: Liens on page 1862](#)
- [Resources introduction on page 1446](#)

## Accounting Organizations

This section describes the services available through Moab Web Services for interacting with the **Organization** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.



The [Fields: Organizations](#) reference contains the type and description of the default fields for the **Organization** object.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/organizations</a>	<a href="#">Get all organizations</a>	--	--	--
<a href="#">/rest/accounting/organizations/&lt;id&gt;</a>	<a href="#">Get single organization</a>	--	--	--

This topic contains these sections:

- [Getting organizations on page 1478](#)
  - [Get all organizations on page 1478](#)
  - [Get single organization on page 1479](#)

## Getting organizations

The HTTP GET method is used to retrieve **Organizations** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/organizations?api-version=3
GET http://localhost:8080/mws/rest/accounting/organizations/<id>?api-version=3
```

## Get all organizations

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/organizations?api-version=3&proxy-user=<user>[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]][&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>query</b>	No	JSON	--	Results are restricted to those having the specified field values. <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  The query parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.                 </div>	query={"deleted": -false}
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	sort={"requestedId": -1}
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/organizations?api-version=3&proxy-user=moab&fields=id,description&sort={"id":1}&pretty=true
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "description": "Arts College",
      "id": "arts"
    },
    {
      "description": "Sciences College",
      "id": "sciences"
    }
  ]
}
```

**Get single organization**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/organizations/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	--
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all-1=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/organizations/sciences?api-version=3&proxy-user=moab&pretty=true
-----
{
  "description": "Sciences College",
  "id": "sciences"
}
```

Related topics

- [Fields: Organizations](#) on page 1868
- [Resources introduction](#) on page 1446

## Accounting Quotes

This section describes the services available through Moab Web Services for interacting with the **Quote** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Quotes](#) reference contains the type and description of the default fields for the **Quotes** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/quotes</a>	<a href="#">Get all quotes</a>	--	--	--
<a href="#">/rest/accounting/quotes/&lt;id&gt;</a>	<a href="#">Get single quote</a>	--	--	--

This topic contains these sections:

- [Getting quotes](#) on page 1480
  - [Get all quotes](#) on page 1480
  - [Get single quote](#) on page 1482

## Getting quotes

The HTTP GET method is used to retrieve **Quote** information.

Quick reference

```
GET http://localhost:8080/mws/rest/accounting/quotes?api-version=3
GET http://localhost:8080/mws/rest/accounting/quotes/<id>?api-version=3
```

### Get all quotes

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/quotes?api-version=3&proxy-user=<user>
[&active=true][&filter=<filter_options>[&filter-type=<filter_type>]][&query=<query_
conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]|&show-all=
(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform actions as defined MAM user.	proxy-user=amy
<b>active</b>	No	Boolean	true or false	Lists only active or non-active quotes.	active=true
<b>filter</b>	No	JSON	--	Query funds based on defined MAM filter.	filter={"account":"chemistry"}
<b>filter-type</b>	No	String	--	Query funds based on defined MAM filter type.	filter-type=NonExclusive
<b>query</b>	No	JSON	--	Results are restricted to those having the specified field values.  <div style="border: 1px solid black; padding: 5px; background-color: #e6f2ff;"> <p><b>i</b> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	query={"instance":"-job.1"}
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,instance,amount

Parameter	Required	Type	Valid values	Description	Example
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	sort={"instance":1}
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/quotes?api-version=3&proxy-user=amy&filter={"account":"chemistry"}&fields=usageRecord,amount&active=true&pretty=true
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "usageRecord": 1,
    "amount": 57600
  } ]
}
```

**Get single quote**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/quotes/<id>?api-version=3&proxy-user=<user>[&active=(true|false)] [&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	--

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>active</b>	No	Boolean	true or false	Lists only active or non-active quotes.	active=true
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id,name,amount
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/accounting/quotes/1?api-version=3&proxy-user=amy&pretty=true
```

```
{
  "id": 1,
  "amount": 57600,
  "pinned": true,
  "instance": "",
  "usageRecord": 1,
  "startTime": "2013-08-21 16:45:57 UTC",
  "endTime": "2013-08-21 17:57:57 UTC",
  "duration": 3600,
  "description": "",
  "chargeRates": [ {
    "name": "Processors",
    "value": "",
    "amount": "1/s"
  } ]
}
```

#### Related topics

- [Fields: Quotes on page 1871](#)
- [Resources introduction on page 1446](#)

## Accounting Transactions

This section describes the services available through Moab Web Services for interacting with the **Transaction** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses

delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Transactions](#) reference contains the type and description of the default fields for the **Transaction** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/accounting/transactions</a>	<a href="#">Get all transactions</a>	--	--	--
<a href="#">/rest/accounting/transactions/&lt;id&gt;</a>	<a href="#">Get single transaction</a>	--	--	--

This topic contains these sections:

- [Getting transactions on page 1484](#)
  - [Get all transactions on page 1484](#)
  - [Get single transaction on page 1487](#)

## Getting transactions

The HTTP GET method is used to retrieve **Transaction** information.

Quick reference

```
GET http://localhost:8080/mws/rest/accounting/transactions?api-version=3
GET http://localhost:8080/mws/rest/accounting/transactions/<id>?api-version=3
```

## Get all transactions

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/transactions?api-version=3&proxy-user=<user>[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>][&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy

Parameter	Required	Type	Valid values	Description	Example
<b>query</b>	No	JSON	--	<p>Results are restricted to those having the specified field values.</p> <div style="border: 1px solid #004a7c; padding: 5px; background-color: #e6f2ff;"> <p> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	<code>query={"action":"Charge","account":"chemistry"}</code>
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	<code>fields=id</code>

Parameter	Required	Type	Valid values	Description	Example
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">Accounting Transactions</a> parameter.	<code>sort={"id":1}</code>
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	<code>show-all=true</code>

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/transactions?api-version=3&proxy-user=moab&query={"instance":"job.1"}&fields=object,action,instance,amount&pretty=true
-----
{
  "totalCount": 310,
  "resultCount": 3,
  "results": [
    {
      "object": "UsageRecord",
      "action": "Reserve",
      "instance": "job.1",
      "amount": 57600
    },
    {
      "object": "UsageRecord",
      "action": "Charge",
      "instance": "job.1",
      "amount": 11520
    },
    {
      "object": "UsageRecord",
      "action": "Refund",
      "instance": "job.1",
      "amount": 11520
    }
  ]
}
```

**Get single transaction**

URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/transactions/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	--
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=id
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all-1=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/transactions/1?api-version=3&proxy-user=moab&pretty=true
-----
{
  "id": 1,
  "object": "Organization",
  "action": "Create",
  "actor": "scottmo",
  "key": "sciences",
  "child": "",
  "count": 1,
  "instance": "",
  "amount": "",
  "delta": "",
  "user": "",
  "account": "",
  "machine": "",
  "fund": "",
  "allocation": "",
  "usageRecord": "",
  "duration": "",
  "description": ""
}
```

Related topics

- [Fields: Transactions on page 1878](#)
- [Resources introduction on page 1446](#)

## Accounting Usage records

This section describes the services available through Moab Web Services for interacting with the **Usage Record** object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Usage Records](#) reference section contains the type and description of all fields in the **Usage Record** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/accounting/usage-records</b>	<a href="#">Get all usage records</a>	--	--	--
<b>/rest/accounting/usage-records/&lt;id&gt;</b>	<a href="#">Get single usage record</a>	--	--	--
<b>/rest/accounting/usage-records/quote</b>	--	--	<a href="#">Obtain a quote for resource usage</a>	--

This topic contains these sections:

- [Getting usage records on page 1489](#)
  - [Get all usage records on page 1489](#)
  - [Get single usage record on page 1492](#)
  - [Obtain a quote for resource usage on page 1493](#)

## Getting usage records

The HTTP GET method is used to retrieve **Usage Record** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/usage-records?api-version=3
GET http://localhost:8080/mws/rest/accounting/usage-records/<id>?api-version=3
POST http://localhost:8080/mws/rest/accounting/usage-records/quote?api-version=3
```

### Get all usage records

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/usage-records?api-version=3&proxy-user=<user>[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy

Parameter	Required	Type	Valid values	Description	Example
<b>query</b>	No	JSON	--	<p>Results are restricted to those having the specified field values.</p> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p> The <b>query</b> parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.</p> </div>	<code>query={"account":"query"}</code>
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	<code>fields=id,instance,charge,user,account</code>

Parameter	Required	Type	Valid values	Description	Example
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	sort={"user":1}
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/accounting/usage-records?api-version=3&proxy-user=amy&fields=id,instance,charge,user,account&pretty=true
```

```
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "id": 1,
      "instance": "job.1",
      "charge": 31,
      "user": "amy",
      "account": "chemistry"
    },
    {
      "id": 2,
      "instance": "job.2",
      "charge": 30,
      "user": "amy",
      "account": "biology"
    }
  ]
}
```

## Get single usage record

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/usage-records/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object.	code
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	field-s=id,instance,charge,user,account
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/usage-records/1?api-version=3&proxy-user=amy&pretty=true
-----
{
  "id": 1,
  "type": "Job",
  "instance": "job.1",
  "charge": 31,
  "stage": "Charge",
  "user": "amy",
  "group": "faculty",
  "account": "chemistry",
  "organization": "sciences",
  "qualityOfService": "",
  "machine": "colony",
  "nodes": "",
  "processors": 16,
  "memory": "",
  "disk": "",
  "network": "",
  "duration": 720,
  "startTime": "",
  "endTime": "",
  "description": ""
}
```

**Obtain a quote for resource usage**

URLs and parameters

```
POST http://localhost:8080/mws/rest/accounting/usage-records/quote?api-version=3&object-type=<object>&proxy-user=<user>&charge-duration=<seconds>
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>charge-duration</b>	Yes	Integer	--	The quote duration of the job in seconds.	charge-duration=6400

Parameter	Required	Type	Valid values	Description	Example
<b>object-type</b>	Yes	String	--	The object to quote. It can be job or service.	<code>object-type=job</code>
<b>itemize</b>	No	Boolean	true or false	Returns the composite charge information in the response data.	<code>itemize=true</code>
<b>rate</b>	No	JSONArray	--	Uses the specified charge rates in the quote. The specified rates override the standard and quote rates. If the guarantee field is set to true, these charge rates will be saved and used when this quote is referenced in a charge action.	<code>rate=[{"type":"VBR","name":"Memory","rate":1}, {"type":"VBR","name":"Processors","rate":1}]</code>

Parameter	Required	Type	Valid values	Description	Example
<b>guarantee</b>	No	Boolean	true or false	Guarantees the quote and returns a quote id to secure the current charge rates. This results in the creation of a quote record and a permanent usage record. This parameter is mutually exclusive with the cost-only parameter.	guarantee=true

Parameter	Required	Type	Valid values	Description	Example
<b>grace-duration</b>	No	Integer	--	The guaranteed quote grace period in seconds. If the quote duration is specified but not the quote end time, the quote endtime will be calculated as the quote start time plus the quote duration plus the grace duration.	grace-duration=6400
<b>cost-only</b>	No	Integer	--	Returns the cost, ignoring all balance and validity checks. This parameter is mutually exclusive with the guarantee parameter.	cost-only=true

Parameter	Required	Type	Valid values	Description	Example
<b>description</b>	No	String	--	The guaranteed quote description.	description="ABC Coupon Rate"
<b>start-time</b>	No	Date	--	The guaranteed quote start time in the format yyyy-MM-dd HH:m-m:ss z,-Infinity,Infinity, or Now.	start-time="2012-04-09 13:49:40 UTC"
<b>end-time</b>	No	Date	--	The guaranteed quote end time in the format yyyy-MM-dd HH:m-m:ss z,-Infinity,Infinity, or Now.	end-time="2012-04-09 14:49:40 UTC"

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

The request body below shows all of the fields in a job that could affect the quote.

```
POST http://localhost:8080/mws/rest/accounting/usage-records/quote?api-  
version=3&object-type=job&charge-duration=300
```

```
{  
  "id": "Moab.1",  
  "user": "amy",  
  "group": "group",  
  "rmName": "machine1",  
  "templateList": [  
    "genericVm"  
  ],  
  "account": "biology",  
  "qosRequested": "QOS1",  
  "variables": {  
    "imageName": "centos5.5-stateless",  
    "topLevelServiceId": "myService.1",  
    "serviceId": "vmService.1",  
    "vmid": "VmService.1",  
    "pmid": "VmService.1"  
  },  
  "requirements": [  
    {  
      "requiredProcessorsPerTask": 2,  
      "genericResources": {  
        "gold": 100,  
        "os": 500  
      },  
      "requiredNodeCountMinimum": 1,  
      "requiredMemoryPerTask": 1024,  
      "requiredClass": "batch"  
    }  
  ]  
}
```

The request body below shows all of the fields in a service that affect the quote in a default MAM installation.

```
POST http://localhost:8080/mws/rest/accounting/usage-records/quote?api-
version=3&object-type=service&charge-duration=300
```

```
{
  "name": "service.1",
  "user": "amy",
  "account": "chemistry"
  "attributes": {
    "moab": {
      "job": {
        "resources": {
          "procs": 1,
          "mem": 2048,
          "OS": 500,
          "gold": 100
        },
        "variables": {
          "Var1": 1524
        },
        "image": "centos5.5-stateless",
        "template": "genericVM",
      }
    }
  }
}
```

### Sample response

- If the quote is not guaranteed:

JSON response

```
{
  "instance": "Moab.1",
  "amount": 600
}
```

- If the quote is guaranteed:

JSON response

```
{
  "id": 1,
  "usageRecord": 2,
  "instance": "Moab.1",
  "amount": 600
}
```

- If the quote is guaranteed and itemized:

JSON response

```
{
  "details": [
    {
      "name": "Processors",
      "value": "2",
      "duration": 300,
      "rate": 1,
      "scalingFactor": 1,
      "amount": 600,
      "details": "2 [Processors] * 1 [ChargeRate{VBR}{Processors}] * 300
[Duration]"
    },
    {
      "name": "Memory",
      "value": "1024",
      "duration": 300,
      "rate": 1,
      "scalingFactor": 1,
      "amount": 307200,
      "details": "1024 [Memory] * 1 [ChargeRate{VBR}{Memory}] * 300 [Duration]"
    }
  ],
  "id": 20,
  "instance": "Moab.1",
  "usageRecord": 20,
  "amount": 307800
}
```

- If the quote is on a service:

JSON response

```

{
  "services": [
    {
      "details": [
        {
          "name": "Processors",
          "value": "22",
          "duration": 30,
          "rate": 1,
          "scalingFactor": 1,
          "amount": 660,
          "details": "22 [Processors] * 1 [ChargeRate{VBR}{Processors}] * 30
[Duration]"
        },
        {
          "name": "Memory",
          "value": "32343242",
          "duration": 30,
          "rate": 1,
          "scalingFactor": 1,
          "amount": 970297260,
          "details": "32343242 [Memory] * 1 [ChargeRate{VBR}{Memory}] * 30
[Duration]"
        }
      ],
      "id": 120,
      "instance": "myVmWorkflow",
      "usageRecord": 157,
      "amount": 970297920
    },
    {
      "details": [
        {
          "name": "Storage",
          "value": "2500",
          "duration": 30,
          "rate": 1.157E-7,
          "scalingFactor": 1,
          "amount": 0,
          "details": "2500 [Storage] * 1.157e-07 [ChargeRate{VBR}{Storage}] * 30
[Duration]"
        }
      ],
      "id": 122,
      "instance": "myExtraStorageWorkflow",
      "usageRecord": 159,
      "amount": 0
    },
    {
      "details": [
        {
          "name": "Processors",
          "value": "0",
          "duration": 30,
          "rate": 1,
          "scalingFactor": 1,
          "amount": 0,
          "details": "0 [Processors] * 1 [ChargeRate{VBR}{Processors}] * 30
[Duration]"
        }
      ],
    }
  ]
}

```

```

    "name": "Memory",
    "value": "0",
    "duration": 30,
    "rate": 1,
    "scalingFactor": 1,
    "amount": 0,
    "details": "0 [Memory] * 1 [ChargeRate{VBR}{Memory}] * 30 [Duration]"
  }
],
"id": 123,
"instance": "myPmWorkflow",
"usageRecord": 160,
"amount": 0
}
],
"amount": 970297920
}

```

### Restrictions

The `details` field is only available with MAM version 7.1.0 or later.

### Related topics

- [Fields: Usage Records on page 1884](#)
- [Resources introduction on page 1446](#)

## Accounting Users

This section describes the services available through Moab Web Services for interacting with the User object in Moab Accounting Manager. It contains the URLs, request bodies, and responses delivered to and from MWS as an intermediary for MAM.

**i** The [Fields: Users](#) reference contains the type and description of all fields in the **User** object.

### Supported methods

Resource	GET	PUT	POST	DELETE
<code>rest/accounting/users</code>	<a href="#">Get all users</a>	--	--	--
<code>rest/accounting/users/&lt;id&gt;</code>	<a href="#">Get single user</a>	--	--	--

This topic contains these sections:

- [Getting users on page 1503](#)
  - [Get all users on page 1503](#)
  - [Get single user on page 1505](#)

## Getting users

The HTTP GET method is used to retrieve **User** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/accounting/users?api-version=3
GET http://localhost:8080/mws/rest/accounting/users/<id>?api-version=3
```

## Get all users

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/users?api-version=3&proxy-user=<user>
[&query=<query_conditions>][&fields=<fields_to_display>[&sort=<fields_to_sort>]|&show-
all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>proxy-user</b>	Yes	String	--	Perform action as defined MAM user.	proxy-user=amy
<b>query</b>	No	JSON	--	Results are restricted to those having the specified field values.	query={"active":true}

**i** The **query** parameter does not support the full Mongo query syntax. Only querying for a simple, non-nested JSON object is allowed.

Parameter	Required	Type	Valid values	Description	Example
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	<code>fields=name,defaultAccount</code>
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending. Should be used in conjunction with the <a href="#">fields</a> parameter.	<code>sort={"defaultAccount":1}</code>
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	<code>show-all=true</code>

See [Global URL parameters on page 1425](#) for available URL parameters.

## Sample response

```
GET http://localhost:8080/mws/rest/accounting/users?api-version=3&proxy-user=moab&query={"active":true}&pretty=true
```

```
{
  "totalCount": 6,
  "resultCount": 4,
  "results": [
    {
      "active": true,
      "commonName": "",
      "phoneNumber": "",
      "emailAddress": "",
      "defaultAccount": "",
      "description": "Accounting Admin",
      "id": "scottmo"
    },
    {
      "active": true,
      "commonName": "Amy Miller",
      "phoneNumber": "(801) 555-1437",
      "emailAddress": "amy@hpc.com",
      "defaultAccount": "chemistry",
      "description": "",
      "id": "amy"
    },
    {
      "active": true,
      "commonName": "Robert Taylor",
      "phoneNumber": "(801) 555-1474",
      "emailAddress": "bob@hpc.com",
      "defaultAccount": "biology",
      "description": "",
      "id": "bob"
    },
    {
      "active": true,
      "commonName": "David Jones",
      "phoneNumber": "(801) 555-1436",
      "emailAddress": "dave@hpc.com",
      "defaultAccount": "film",
      "description": "",
      "id": "dave"
    }
  ]
}
```

## Get single user

### URLs and parameters

```
GET http://localhost:8080/mws/rest/accounting/users/<id>?api-version=3&proxy-user=<user>[&fields=<fields_to_display>|&show-all=(true|false)]
```

Parameter	Required	Type	Valid values	Description	Example
<b>id</b>	Yes	String	--	The unique identifier of the object	--
<b>fields</b>	No	String	--	Comma-separated list of field names to display.	fields=name,defaultAccount
<b>show-all</b>	No	Boolean	true or false	true shows all fields including metadata and hidden fields. Default is false.	show-all=true

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/accounting/users/amy?api-version=3&proxy-user=moab&pretty=true
-----
{
  "active": true,
  "commonName": "Amy Miller",
  "phoneNumber": "(801) 555-1437",
  "emailAddress": "amy@hpc.com",
  "defaultAccount": "chemistry",
  "description": "",
  "id": "amy"
}
```

Related topics

- [Fields: Users on page 1890](#)
- [Resources introduction on page 1446](#)

## Credentials

This section describes behavior of the **Credential** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

 The Credential API is new with *API version 2*. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3`.

For more information, see [Requesting specific API versions on page 1428](#).

**i** The [Fields: Credentials](#) reference contains the type and description of all fields in the **Credential** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/credentials/accounts</a>	<a href="#">Get all account credentials</a> <a href="#">Get single account credential</a>	<a href="#">Modify account credentials on page 1519</a>	--	--
<a href="#">/rest/credentials/classes</a>	<a href="#">Get all class credentials</a> <a href="#">Get single class credential</a>	<a href="#">Modify class credentials on page 1519</a>	--	--
<a href="#">/rest/credentials/groups</a>	<a href="#">Get all group credentials</a> <a href="#">Get single group credential</a>	<a href="#">Modify group credentials on page 1520</a>	--	--
<a href="#">/rest/credentials/qoses</a>	<a href="#">Get all QoS credentials</a> <a href="#">Get single QoS credential</a>	<a href="#">Modify QoS credentials on page 1520</a>	--	--
<a href="#">/rest/credentials/users</a>	<a href="#">Get all user credentials</a> <a href="#">Get single user credential</a>	<a href="#">Modify User credentials on page 1520</a>	--	--

This topic contains these sections:

- [Getting credentials on page 1508](#)
  - [Get all account credentials on page 1508](#)
  - [Get single account credential on page 1509](#)
  - [Get all class credentials on page 1510](#)
  - [Get single class credential on page 1511](#)
  - [Get all group credentials on page 1512](#)
  - [Get single group credential on page 1513](#)

- [Get all QoS credentials on page 1514](#)
- [Get single QoS credential on page 1515](#)
- [Get all user credentials on page 1516](#)
- [Get single user credential on page 1517](#)
- [Modifying credentials on page 1518](#)
  - [Modify account credentials on page 1519](#)
  - [Modify class credentials on page 1519](#)
  - [Modify group credentials on page 1520](#)
  - [Modify QoS credentials on page 1520](#)
  - [Modify User credentials on page 1520](#)

## Getting credentials

The HTTP GET method is used to retrieve **Resource Type** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/credentials/accounts[/<name>]?api-version=3
GET http://localhost:8080/mws/rest/credentials/classes[/<name>]?api-version=3
GET http://localhost:8080/mws/rest/credentials/groups[/<name>]?api-version=3
GET http://localhost:8080/mws/rest/credentials/qoses[/<name>]?api-version=3
GET http://localhost:8080/mws/rest/credentials/users[/<name>]?api-version=3
```

## Get all account credentials

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/accounts?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/accounts?api-version=3
```

```

{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {
      "name": "Administration",
      "account_access_list": ["Administration"],
      "default_account": "Administration",
      "qos_access_list": [
        "qos1",
        "qos2"
      ],
      "default_qos": "qos1",
      "partition_access_list": [
        "partition1",
        "SHARED"
      ],
      "default_partition": "partition1",
      "target_type": "CEILING",
      "target": 18.43,
      "priority": 53,
      "max_job_duration_in_seconds": 234,
      "max_idle_jobs": 42,
      "max_jobs": 523,
      "max_processors": 4,
      "max_processor_seconds": 525,
      "max_nodes": 75,
      "reservation": "system.1",
      "user_access_list": ["adaptive"]
    }
  ]
}

```

### Get single account credential

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/accounts/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/accounts/Administration?api-version=3
```

```
{
  "name": "Administration",
  "account_access_list": ["Administration"],
  "default_account": "Administration",
  "qos_access_list": [
    "qos1",
    "qos2"
  ],
  "default_qos": "qos1",
  "partition_access_list": [
    "partition1",
    "SHARED"
  ],
  "default_partition": "partition1",
  "target_type": "CEILING",
  "target": 18.43,
  "priority": 53,
  "max_job_duration_in_seconds": 234,
  "max_idle_jobs": 42,
  "max_jobs": 523,
  "max_processors": 4,
  "max_processor_seconds": 525,
  "max_nodes": 75,
  "reservation": "system.1",
  "user_access_list": ["adaptive"]
}
```

## Get all class credentials

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/classes?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/classes?api-version=3
```

```
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {
      "name": "highprio",
      "account_access_list": ["Administration"],
      "default_account": "Administration",
      "qos_access_list": [
        "qos1",
        "qos2"
      ],
      "default_qos": "qos1",
      "partition_access_list": [
        "partition1",
        "SHARED"
      ],
      "default_partition": "partition1",
      "target_type": "CEILING",
      "target": 18.43,
      "priority": 53,
      "max_job_duration_in_seconds": 234,
      "max_idle_jobs": 42,
      "max_jobs": 523,
      "max_processors": 4,
      "max_processor_seconds": 525,
      "max_nodes": 75,
      "reservation": "system.1",
      "user_access_list": ["adaptive"]
    }
  ]
}
```

### Get single class credential

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/classes/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/classes/highprio?api-version=3
```

```
{
  "name": "highprio",
  "account_access_list": ["Administration"],
  "default_account": "Administration",
  "qos_access_list": [
    "qos1",
    "qos2"
  ],
  "default_qos": "qos1",
  "partition_access_list": [
    "partition1",
    "SHARED"
  ],
  "default_partition": "partition1",
  "target_type": "CEILING",
  "target": 18.43,
  "priority": 53,
  "max_job_duration_in_seconds": 234,
  "max_idle_jobs": 42,
  "max_jobs": 523,
  "max_processors": 4,
  "max_processor_seconds": 525,
  "max_nodes": 75,
  "reservation": "system.1",
  "user_access_list": ["adaptive"]
}
```

## Get all group credentials

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/groups/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/groups?api-version=3
```

```

{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {
      "name": "students",
      "account_access_list": ["Administration"],
      "default_account": "Administration",
      "qos_access_list": [
        "qos1",
        "qos2"
      ],
      "default_qos": "qos1",
      "partition_access_list": [
        "partition1",
        "SHARED"
      ],
      "default_partition": "partition1",
      "target_type": "CEILING",
      "target": 18.43,
      "priority": 53,
      "max_job_duration_in_seconds": 234,
      "max_idle_jobs": 42,
      "max_jobs": 523,
      "max_processors": 4,
      "max_processor_seconds": 525,
      "max_nodes": 75,
      "reservation": "system.1",
      "user_access_list": ["adaptive"]
    }
  ]
}

```

## Get single group credential

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/groups/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/groups/students?api-version=3
```

```

{
  "name": "students",
  "account_access_list": ["Administration"],
  "default_account": "Administration",
  "qos_access_list": [
    "qos1",
    "qos2"
  ],
  "default_qos": "qos1",
  "partition_access_list": [
    "partition1",
    "SHARED"
  ],
  "default_partition": "partition1",
  "target_type": "CEILING",
  "target": 18.43,
  "priority": 53,
  "max_job_duration_in_seconds": 234,
  "max_idle_jobs": 42,
  "max_jobs": 523,
  "max_processors": 4,
  "max_processor_seconds": 525,
  "max_nodes": 75,
  "reservation": "system.1",
  "user_access_list": ["adaptive"]
}

```

## Get all QoS credentials

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/qoses?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/qoses?api-version=3
```

```

{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {
      "name": "special",
      "account_access_list": ["Administration"],
      "default_account": "Administration",
      "qos_access_list": [
        "qos1",
        "qos2"
      ],
      "default_qos": "qos1",
      "partition_access_list": [
        "partition1",
        "SHARED"
      ],
      "default_partition": "partition1",
      "target_type": "CEILING",
      "target": 18.43,
      "priority": 53,
      "max_job_duration_in_seconds": 234,
      "max_idle_jobs": 42,
      "max_jobs": 523,
      "max_processors": 4,
      "max_processor_seconds": 525,
      "max_nodes": 75,
      "reservation": "system.1",
      "user_access_list": ["adaptive"],
      "flags": [
        "DEADLINE",
        "RESERVEALWAYS",
        "DEDICATED"
      ]
    }
  ]
}

```

## Get single QoS credential

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/qoses/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/qoses/special?api-version=3
```

```
{
  "name": "special",
  "account_access_list": ["Administration"],
  "default_account": "Administration",
  "qos_access_list": [
    "qos1",
    "qos2"
  ],
  "default_qos": "qos1",
  "partition_access_list": [
    "partition1",
    "SHARED"
  ],
  "default_partition": "partition1",
  "target_type": "CEILING",
  "target": 18.43,
  "priority": 53,
  "max_job_duration_in_seconds": 234,
  "max_idle_jobs": 42,
  "max_jobs": 523,
  "max_processors": 4,
  "max_processor_seconds": 525,
  "max_nodes": 75,
  "reservation": "system.1",
  "user_access_list": ["adaptive"]
  "flags": [
    "DEADLINE",
    "RESERVEALWAYS",
    "DEDICATED"
  ]
  "queue_time_weight": 30,
  "expansion_factor_weight": 40,
  "quality_of_service_priority": 20
}
```

## Get all user credentials

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/users?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/users?api-version=3
```

```

{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {
      "name": "root",
      "account_access_list": ["Administration"],
      "default_account": "Administration",
      "qos_access_list": [
        "qos1",
        "qos2"
      ],
      "default_qos": "qos1",
      "partition_access_list": [
        "partition1",
        "SHARED"
      ],
      "default_partition": "partition1",
      "target_type": "CEILING",
      "target": 18.43,
      "priority": 53,
      "max_job_duration_in_seconds": 234,
      "max_idle_jobs": 42,
      "max_jobs": 523,
      "max_processors": 4,
      "max_processor_seconds": 525,
      "max_nodes": 75,
      "email": "root@root.com"
    }
  ]
}

```

## Get single user credential

### URLs and parameters

```
GET http://localhost:8080/mws/rest/credentials/users/<name>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/credentials/users/root?api-version=3
```

```
{
  "name": "root",
  "account_access_list": ["Administration"],
  "default_account": "Administration",
  "qos_access_list": [
    "qos1",
    "qos2"
  ],
  "default_qos": "qos1",
  "partition_access_list": [
    "partition1",
    "SHARED"
  ],
  "default_partition": "partition1",
  "target_type": "CEILING",
  "target": 18.43,
  "priority": 53,
  "max_job_duration_in_seconds": 234,
  "max_idle_jobs": 42,
  "max_jobs": 523,
  "max_processors": 4,
  "max_processor_seconds": 525,
  "max_nodes": 75,
  "email": "root@root.com"
}
```

## Modifying credentials

The HTTP PUT method is used to modify **credentials**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/credentials/accounts/<name>?api-version=3 [&change-
mode=<add|remove|set>]
PUT http://localhost:8080/mws/rest/credentials/classes/<name>?api-version=3 [&change-
mode=<add|remove|set>]
PUT http://localhost:8080/mws/rest/credentials/groups/<name>?api-version=3 [&change-
mode=<add|remove|set>]
PUT http://localhost:8080/mws/rest/credentials/qoses/<name>?api-version=3 [&change-
mode=<add|remove|set>]
PUT http://localhost:8080/mws/rest/credentials/users/<name>?api-version=3 [&change-
mode=<add|remove|set>]
```

### URL parameters

URL parameters for modifying a credential.

Credentials parameter	Required	Type	Valid values	Description
<b>change-mode</b>	No	String	set (default) add remove	If <b>set</b> , replace existing list with the given one. If <b>add</b> , add the given field(s) to the existing list. If <b>remove</b> , remove the given field(s) from the existing list.

**i** Moab Workload Manager will automatically add SHARED and the value of default\_partition to the partition\_access\_list.

## Modify account credentials

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/credentials/accounts/<name>?api-version=3 [&change-mode=<add|remove|set>]
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/credentials/accounts/biology?api-version=3&change-mode=add
-----
{
  "qos_access_list": [
    "qos3",
    "qos4"
  ],
  "max_job_duration_in_seconds": 234
}
```

## Modify class credentials

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/credentials/classes/<name>?api-version=3 [&change-mode=<add|remove|set>]
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/credentials/classes/highprio?api-version=3
-----
{
  "max_idle_jobs": 50,
  "max_jobs": 300
}
```

## Modify group credentials

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/credentials/groups/<name>?api-version=3 [&change-
mode=<add|remove|set>]
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/credentials/groups/students?api-version=3&change-
mode=set
-----
{
  "reservation": "system.2",
  "user_access_list": ["tom"]
}
```

## Modify QoS credentials

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/credentials/qoses/<name>?api-version=3 [&change-
mode=<add|remove|set>]
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/credentials/qoses/special?api-version=3
-----
{
  "max_processors": 5,
  "max_processor_seconds": 500
}
```

## Modify User credentials

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/credentials/users/<name>?api-version=3 [&change-
mode=<add|remove|set>]
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/credentials/users/tom?api-version=3
```

```
{
  "email": "tom@root.com"
}
```

### Related topics

- [Fields: Credentials on page 1893](#)
- [Resources introduction on page 1446](#)

## Diagnostics

This section describes additional REST calls that are available for performing diagnostics on Moab Web Services.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/diag/about</a>	<a href="#">Get version information</a>	--	--	--
<a href="#">/rest/diag/auth</a>	<a href="#">Diagnose authentication</a>	--	--	--
<a href="#">/rest/diag/health/summary</a>	<a href="#">Get health summary</a>	--	--	--
<a href="#">/rest/diag/health/detail</a>	<a href="#">Get health detail</a>	--	--	--

**i** [/rest/diag/ldap](#) is deprecated. All information that was available in that resource is now available in [/rest/diag/health/detail](#).

This topic contains these sections:

- [Get version information on page 1522](#)
- [Diagnose authentication on page 1522](#)
- [Connection health information on page 1522](#)
  - [Get health summary on page 1523](#)
  - [Get health detail on page 1523](#)
- [Diagnostics on page 1521](#)

## Get version information

The HTTP GET method is used to retrieve version and build information.

### [Quick reference](#)

```
GET http://localhost:8080/mws/rest/diag/about?api-version=3
```

### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/diag/about?api-version=3
```

### [Sample response](#)

The response contains the application suite, version, build date, and revision.

```
{
  "suite": "CLOUD",
  "version": "7.2.2",
  "buildDate": "2013.03.15_13.12.45",
  "revision": "302238e24e327f4aa45ab4c91834216a7fc19d63"
}
```

## Diagnose authentication

The HTTP GET method is used to test for proper authentication. This resource is designed to be used as a simple validation of credentials and gives no output besides the response code.

### [Quick reference](#)

```
GET http://localhost:8080/mws/rest/diag/auth?api-version=3
```

### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/diag/auth?api-version=3
```

### [Sample response](#)

 A successful result is indicated by the 200 response code while a failure is indicated by a 401 response code.

```
{}
```

## Connection health information

The HTTP GET method is used to retrieve health or status information for connections to external systems or software. There are two available resources for health, one that returns simple summary information and another that returns detailed information.

### Quick reference

```
GET http://localhost:8080/mws/rest/diag/health/summary?api-version=3
GET http://localhost:8080/mws/rest/diag/health/detail?api-version=3
```

## Get health summary

### URLs and parameters

```
GET http://localhost:8080/mws/rest/diag/health/summary?api-version=3
```

**i** If the MongoDB connection is down, authenticated resources are not available. While this resource does not possess much detail beyond that of simple connection information, it is still useful as it does not require authentication and therefore can be used to determine connection problems with MongoDB.

### Sample response

The response contains the connection health for Moab Workload Manager (MWM), Moab Accounting Manager (MAM), MongoDB, LDAP, ZeroMQ, PAM. A `true` response value indicates that the connection is healthy and available, and a `false` response indicates that the connection is currently down. Likewise, the `mongoConnected` property for Moab signifies the state of the Moab to MongoDB connection. The possible values of this state are `UP`, `DOWN`, `NOT_CONFIGURED` (when the MongoDB server is not configured in Moab), `NOT_SUPPORTED` (when Moab is not compiled with MongoDB support), and `UNKNOWN` (when MWS cannot communicate with Moab).

```
{
  "mam": {"connected": true},
  "mongo": {"connected": true},
  "mwm": {
    "connected": true,
    "mongoConnected": "UP",
    "zmqConnected": true
  },
  "ldap": {"connected": true},
  "pam": {"connected": true},
  "zmq": {"connected": true},
}
```

## Get health detail

### URLs and parameters

```
GET http://localhost:8080/mws/rest/diag/health/detail?api-version=3
```

**i** If the MongoDB connection is down, authenticated resources such as this are not available. In this case, using the [Get health summary](#) instead may be required.

### Sample response

The response contains the connection health and information for Moab Workload Manager (MWM), Moab Accounting Manager (MAM), MongoDB, LDAP, ZeroMQ, PAM. A "connected": true response value indicates that the connection is healthy and available, and a false response indicates that the connection is currently down. Likewise, the mongoConnected property for Moab signifies the state of the Moab to MongoDB connection. The possible values of this state are UP, DOWN, NOT\_CONFIGURED (when the MongoDB server is not configured in Moab), NOT\_SUPPORTED (when Moab is not compiled with MongoDB support), and UNKNOWN (when MWS cannot communicate with Moab). A message is also present for all down connections except Moab to MongoDB giving a reason for the error state.

```

{
  "mam": {
    "connected": true,
    "adminUser": "root",
    "host": "mamhost",
    "port": 7741,
    "version": "7.5",
    "message": null
  },
  "mongo": {
    "connected": true,
    "host": "127.0.0.1",
    "port": 27017,
    "replicaSet": null,
    "databaseName": "mws",
    "username": {},
    "version": "2.4.8",
    "message": null
  },
  "mwm": {
    "connected": true,
    "adminUser": "root",
    "host": "localhost",
    "port": 42559,
    "version": "7.5",
    "licensedFeatures": [
      "green",
      "provision",
      "vm"
    ],
    "state": "RUNNING",
    "mongo": {
      "connected": "UP",
      "credentialsSet": false,
      "host": "myhost",
      "port": 27017
    }
  },
  "zmq": {
    "connected": true,
    "encryptionStatus": "OFF",
    "port": 5563
  },
  "message": null
},
"ldap": {
  "connected": true,
  "message": null,
  "server": "openldapnis.ac",
  "port": 389,
  "baseDNs": ["dc=testldap,dc=ac"],
  "bindUser": "cn=admin,dc=testldap,dc=ac",
  "directoryType": "OpenLDAP Using InetOrgPerson Schema",
  "securityType": "NONE",
  "userObjectClass": "inetOrgPerson",
  "groupObjectClass": "groupOfNames",
  "ouObjectClass": "organizationalUnit",
  "userMembershipAttribute": null,
  "groupMembershipAttribute": "member",
  "userNameAttribute": "uid"
},
"pam": {
  "connected": true,

```

```

    "authenticationModule": "system-auth",
    "message": "PAM is configured in MWS."
  },
  "zmq": {
    "connected": true,
    "version": "3.2.3",
    "message": null,
    "mwmSubscriber": {
      "connected": true,
      "address": "localhost",
      "port": 5563,
      "message": null
    },
    "mwsSubscriber": {
      "connected": true,
      "address": "localhost",
      "port": 5564,
      "message": null
    },
    "publisher": {
      "connected": true,
      "address": "*",
      "port": 5564,
      "message": null
    }
  }
},
}

```

Related topics

- [Resources introduction on page 1446](#)

## Distinct

The **Distinct** resource enables clients to retrieve distinct (unique) values from another MWS resource. For example, a client can request the list of all `featuresReported` across all nodes like this:

```
GET http://localhost:8080/mws/rest/distinct/nodes/featuresReported/?api-version=3
```

Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/distinct/&lt;resource&gt;/&lt;field&gt;</code>	<a href="#">Get distinct values</a>	--	--	--

This topic contains these sections:

- [Get distinct values on page 1526](#)

### Get distinct values

The HTTP GET method is used to retrieve **distinct** values from another MWS resource.

## URLs and parameters

```
GET http://localhost:8080/mws/rest/distinct/<resource>/<field>?api-version=3
```

Parameter	Required	Type	Valid values	Example
<b>resource</b>	Yes	String	The MWS resource to query.	nodes
<b>field</b>	Yes	String	The field for which to return the distinct values.	featuresReported
<b>query</b>	No	JSON	Determines the subset of objects from which to retrieve the distinct values.	query={"states.-powerState": "On"}

**i** The **Distinct** resource has no access control of its own. Rather, it depends on the access control of the MWS resource being queried.

For example, for a client to run a query like `/rest/distinct/nodes/featuresReported`, it must have GET rights on the Nodes resource. For more information, see [Access control on page 1420](#).

## Examples

*Example 4-3: Get all featuresReported across all nodes*

```
http://localhost:8080/mws/rest/distinct/nodes/featuresReported?api-version=3
```

```
{
  "totalCount": 1,
  "resultCount": 1,
  "results": ["vlan1"]
}
```

*Example 4-4: Get all available operating system images across all nodes that are powered on*

```
http://localhost:8080/mws/rest/distinct/nodes/operatingSystem.imagesAvailable?api-version=3&query={"states.powerState": "On"}
```

```
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    "linux",
    "windows"
  ]
}
```

## Related topics

- [Resources introduction on page 1446](#)

# Events

This section describes the URLs, request bodies, and responses delivered to and from Moab Web Services for handling events.

 The Event API is new with API version 3. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3` in order to behave as documented.

For more information, see [Requesting specific API versions on page 1428](#).

 The [Fields: Events](#) reference contains the type and description of all fields in the **Event** object. It also contains details regarding which fields are valid during POST actions.

## Important changes

- The following fields have been renamed in API version 3:

Name in version 1 & 2	Name in version 3
eventTime	eventDate
sourceComponent	origin
errorMessage.message	message
relatedObjects	associatedObjects

- The following fields have been removed in API version 3.

 MWS will no longer report these fields, even if there are existing events in the database with these fields.

- eventCategory
- status
- facility
- initiatedBy
- primaryObject (Primary objects are now reported in associatedObjects.)
- errorMessage.originator
- errorMessage.errorCode
- details

- The following fields are new in API version 3 (see [Fields: Events on page 1894](#)):
  - arguments
  - code

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/events</a>	<a href="#">Get all events</a>	--	<a href="#">Create event</a>	--
<a href="#">/rest/events/&lt;id&gt;</a>	<a href="#">Get single event</a>	--	--	--

This topic contains these sections:

- [Getting events on page 1529](#)
  - [Get all events on page 1529](#)
  - [Get single event on page 1533](#)
- [Creating events on page 1534](#)
  - [Create event on page 1534](#)

## Getting events

The HTTP GET method is used to retrieve **Event** information. Queries for all objects and a single object are available.

Quick reference

```
GET http://localhost:8080/mws/rest/events?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/events/<id>?api-version=3
```

## Get all events

URLs and parameters

```
GET http://localhost:8080/mws/rest/events?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Valid values	Example
<b>query</b>	No	JSON	Query for specific results. It is possible to query events by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"severity":"ERROR"}

Parameter	Required	Type	Valid values	Example
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	<code>sort={"id":-1}</code>

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/events?api-version=3
```

```
{
  "totalCount":2,
  "resultCount":2,
  "results":[
    {
      "arguments":[
      ],
      "associatedObjects":[
        {
          "type":"VM",
          "id":"vml"
        }
      ],
      "tenant":
        {
          "id":"1234567890abcdef12345678",
          "name":"Research"
        },
      "code":234881023,
      "eventDate":"2013-06-10 17:13:31 UTC",
      "eventType":"VM Provision",
      "message":null,
      "origin":"CSA Plugin",
      "severity":"INFO",
      "id":"51b6093bc4aa708a5bebb6ae"
    },
    {
      "arguments":[
        "51b608ddc4aa708a5bebb684"
      ],
      "associatedObjects":[
        {
          "type":"Service",
          "id":"51b608ddc4aa708a5bebb684"
        }
      ],
      "tenant":
        {
          "id":"1234567890abcdef12345678",
          "name":"Research"
        },
      "code":33554944,
      "eventDate":"2013-06-10 17:11:59 UTC",
      "eventType":"Service Create",
      "message":"The service '51b608ddc4aa708a5bebb684' was created",
      "origin":"MWS/ServiceEvents/CREATE_1ID",
      "severity":"INFO",
      "id":"51b608dfc4aa708a5bebb686"
    }
  ]
}
```

Querying events

It is possible to query events by one or more fields based on [MongoDB query syntax](#). The following contains examples of simple and complex event queries and event queries by date.

**Simple queries:**

- To see only events that are of type "Service Create":

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventType":"Service Create"}
```

- To see only events of type "Service Create" with the severity of "INFO":

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventType":"Service Create","severity":"INFO"}
```

- To see only events with a code of 33554946

```
http://localhost:8080/mws/rest/events?api-version=3&query={code:33554946}
```

**More complex queries:**

- You can query on embedded JSON objects within the event JSON. For example, to see events associated with service 51b608ddc4aa708a5bebb684:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"associatedObjects.id":"51b608ddc4aa708a5bebb684"}
```

- To see only events that are NOT associated with service 51b608ddc4aa708a5bebb684:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"associatedObjects.id":{"$ne":"51b608ddc4aa708a5bebb684"}}
```

- When the field values of the desired events are a finite set, you can use the \$in operator. For example, to see events that have a severity of either WARN or ERROR:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"severity":{"$in":["ERROR","WARN"]}}
```

**Querying events by date**

- To see events created before January 27, 2012 at 12:08 a.m. UTC:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventDate":{"$lt":"2012-01-27 12:08:00 UTC"}}
```

- To see events created before or on January 27, 2012 at 12:08 a.m. UTC:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventDate":{"$lte":"2012-01-27 12:08:00 UTC"}}
```

- To see all events created after January 27, 2012 at 12:04 a.m. UTC:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventDate":{"$gt":"2012-01-27 12:04:00 UTC"}}
```

- To see all events created after or on January 27, 2012 at 12:04 a.m. UTC:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventDate":{"$gte":"2012-01-27 12:04:00 UTC"}}
```

- To see events created between 12:04 a.m. and 12:08 a.m. UTC inclusive:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventDate":{"$gte":"2012-01-27 12:04:00 UTC","$lte":"2012-01-27 12:08:00 UTC"}}
```

- To see events created between 12:04 a.m. and 12:08 a.m. UTC inclusive that have a severity of ERROR:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"severity":"ERROR","eventDate":{"$gte":"2012-01-27 12:04:00 UTC","$lte":"2012-01-27 12:08:00 UTC"}}
```

### Sorting

See the sorting section of [Global URL parameters on page 1425](#).

### Limiting the number of results

- If you want to limit the number of results of events, you can use the max parameter. For example, to see only 10 "VM Provision" events:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventType":"VM Provision"}&sort={"eventDate":1}&max=10
```

- To see "VM Provision" events 51-60 when sorted by eventDate in descending order, you can combine max with offset, as follows:

```
http://localhost:8080/mws/rest/events?api-version=3&query={"eventType":"VM Provision"}&sort={"eventDate":-1}&max=10&offset=50
```

## Get single event

### URLs and parameters

```
GET http://localhost:8080/mws/rest/events/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
id	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/events/51b608dfc4aa708a5bebb686?api-version=3
```

```
{
  "arguments": ["51b608ddc4aa708a5bebb684"],
  "associatedObjects": [ {
    "type": "Service",
    "id": "51b608ddc4aa708a5bebb684"
  } ],
  "tenant": {
    "id": "1234567890abcdef12345678",
    "name": "Research"
  },
  "code": 33554944,
  "eventDate": "2013-06-10 17:11:59 UTC",
  "eventType": "Service Create",
  "message": "The service '51b608ddc4aa708a5bebb684' was created",
  "origin": "MWS/ServiceEvents/CREATE_1ID",
  "severity": "INFO",
  "id": "51b608dfc4aa708a5bebb686"
}
```

## Creating events

The HTTP POST method is used to create an **Event**.

### Quick reference

```
POST http://localhost:8080/mws/rest/events?api-version=3
```

### Create event

#### URLs and parameters

```
POST http://localhost:8080/mws/rest/events?api-version=3
```

#### Request body

```
POST http://localhost:8080/mws/rest/events?api-version=3 Content-Type:application/json
```

```
{
  "arguments": ["vm1"],
  "associatedObjects": [ {
    "type": "VM",
    "id": "vm1"
  } ],
  "code": 234881023,
  "eventDate": "2013-06-10 17:13:31 UTC",
  "eventType": "VM Provision",
  "message": "The virtual machine \"vm1\" was provisioned",
  "origin": "CSA Plugin",
  "severity": "INFO"
}
```

**i** An event's `tenant` is automatically inherited from the `associatedObjects`.

### Sample response

If the request was successful, the response will be an object with an `id` property containing the ID of the newly created events. On failure, the response is an error message.

JSON response

```
{ "arguments": ["vm1"], "associatedObjects": [{"_id": "vm1", "id": "vm1", "type": "VM", "version": 0}], "code": 234881023, "eventDate": "2013-06-10 17:13:31 UTC", "eventType": "VM Provision", "id": "51b62046c4aa708a5bebc018", "message": "The virtual machine vm1 was provisioned", "origin": "CSA Plugin", "severity": "INFO", "version": 0 }
```

Below is an example of `events.log` output for a successful event request:

```
2013-06-10T11:13:31.000-06:00 severity="INFO" code="0x0dffffff" type="VM Provision" origin="CSA Plugin" associatedObject.0.type="VM" associatedObject.0.id="vm1" arguments=["vm1"] message="The virtual machine \"vm1\" was provisioned"
```

**i** Note that " (double quote) characters in the input have been replaced by \ " characters in the output. (For other character restrictions, see [Restrictions on page 1535](#).)

### Restrictions

Special characters—such as newline, carriage return, and " (double quote) characters—are encoded in the output of `events.log` to make `events.log` easy to parse with scripts and third party tools. For example, if the input XML contains:

```
<ErrorMessage>RM says, "Cannot provision vm21"</ErrorMessage>
```

Then the following will be output to `events.log`:

```
error.message="RM says, \"Cannot provision vm21\""
```

(Notice that " has been replaced with \ ".)

This table contains the most common encodings. (For more information, see [escape sequences for Java Strings](#).)

Character	Escape sequence
" (double quote)	\ "
\ (backslash)	\\
newline	\n

Character	Escape sequence
carriage return	\r
tab	\t

Other restrictions include:

- **origin**, **eventType**, **associatedObject.id**, and **associatedObject.type** cannot contain single quotes (') or double quotes (").

Related topics

- [Resources introduction on page 1446](#)
- [Notifications on page 1585](#)
- [Fields: Notifications on page 2195](#)
- [Notification conditions on page 1580](#)
- [Fields: Notification Conditions on page 2191](#)
- [Fields: Events on page 1894](#)
- [System events on page 1444](#)
- [Creating events and notifications on page 1704](#)
- [Plugin event service on page 1755](#)
- [Handling events on page 1711](#)
- [Securing the connection with the message queue on page 1417](#)

## Images

This section describes behavior of the **Image** object in Moab Web Services. An image resource is used to track the different types of operating systems and hypervisors available in the data center. It also tracks which virtual machines are available on the hypervisors. This section describes the URLs, request bodies, and responses delivered to and from MWS.

 The [Fields: Images](#) reference contains the type and description of all fields in the **Image** object. It also contains details regarding which fields are valid during PUT and POST actions.

Supported methods

Resource	GET	PUT	POST	DELETE
/rest/images	<a href="#">Get all images</a>	--	<a href="#">Create single image</a>	--

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/images/&lt;id&gt;</a>	<a href="#">Get single image</a>	<a href="#">Modify single image</a>	--	<a href="#">Delete single image</a>
<a href="#">/rest/images/&lt;name&gt;</a>	<a href="#">Get single image</a>	<a href="#">Modify single image</a>	--	<a href="#">Delete single image</a>

This topic contains these sections:

- [Getting images on page 1537](#)
  - [Get all images on page 1537](#)
  - [Get single image on page 1538](#)
- [Creating images on page 1540](#)
  - [Create single image on page 1540](#)
- [Modifying images on page 1543](#)
  - [Modify single image on page 1543](#)
- [Deleting images on page 1544](#)
  - [Delete single image on page 1544](#)

## Getting images

The HTTP GET method is used to retrieve **Image** information. You can query all objects or a single object.

### Quick reference

```
GET http://localhost:8080/mws/rest/images?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/images/<id>?api-version=3
GET http://localhost:8080/mws/rest/images/<name>?api-version=3
```

## Get all images

### URLs and parameters

```
GET http://localhost:8080/mws/rest/images?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query images by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"type":"stateful","osType":"linux"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/images?api-version=3&fields=id,name
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "id": "4fa197e68ca30fc605dd1cf0",
    "name": "centos5-stateful"
  } ]
}
```

Sorting and querying

See the sorting and querying sections of [Global URL parameters on page 1425](#).

**Get single image**

URLs and parameters

```
GET http://localhost:8080/mws/rest/images/<id>?api-version=3
GET http://localhost:8080/mws/rest/images/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Image.
<b>name</b>	Yes	String	--	The name of the Image.

**i** You must specify either **id** or **name**, but you do not have to specify both.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

#### Virtual machine image example:

```
GET http://localhost:8080/mws/rest/images/centos5-compute-stateful?api-version=3
```

```
{
  "active":true,
  "extensions":{
    "xcat":{
      "os":"centos",
      "architecture":"x86_64",
      "profile":"compute"
    }
  },
  "features":[],
  "hypervisor":false,
  "hypervisorType": null,
  "id":"4fa197e68ca30fc605dd1cf0",
  "name":"centos5-compute-stateful",
  "osType":"linux",
  "supportsPhysicalMachine":false,
  "supportsVirtualMachine":true,
  "templateName":null,
  "type":"stateful",
  "version":0,
  "virtualizedImages":[]
}
```

#### Hypervisor image example:

```
GET http://localhost:8080/mws/rest/images/esxi-4.1-stateful?api-version=3
```

```
{
  "active":true,
  "extensions":{
    "xcat":{
      "hvGroupName":"hvGroup",
      "vmGroupName":"vmGroup",
      "os":"esxi-4.1",
      "architecture":"x86_64",
      "profile":"hv"
    }
  },
  "features":[],
  "hypervisor":true,
  "hypervisorType":"ESX",
  "id":"4fa197e68ca30fc605dd1cf0",
  "name":"centos5-compute-stateful",
  "osType":"linux",
  "supportsPhysicalMachine":true,
  "supportsVirtualMachine":false,
  "templateName":null,
  "type":"stateful",
  "version":0,
  "virtualizedImages":[]
}
```

**i** The `version` field contains the current version of the database entry and does not reflect the version of the operating system. For more information, see [Modify single image](#) on page 1543.

## Creating images

The HTTP POST method is used to submit **Images**.

### Quick reference

```
POST http://localhost:8080/mws/rest/images?api-version=3
```

### Create single image

#### URLs and parameters

```
POST http://localhost:8080/mws/rest/images?api-version=3
```

See [Global URL parameters](#) on page 1425 for available URL parameters.

#### Request body

Three fields are required to submit an image: **name**, **hypervisor**, and **osType**. Each image must also support provisioning to either a physical machine or a virtual machine by using the **supportsPhysicalMachine** or **supportsVirtualMachine** fields.

**i** The **name** field must contain only letters, digits, periods, dashes, and underscores.

The array of virtualized images are themselves objects that contain image IDs or names. For more information on available fields and types, see [Fields: Images on page 1903](#).

The following is an example of the most basic image that can be created:

```
POST http://localhost:8080/mws/rest/images?api-version=3
```

```
{
  "name": "centos5-stateful",
  "osType": "linux",
  "hypervisor": false,
  "supportsVirtualMachine": true
}
```

Note that this example does not provide any information for a provisioning manager (such as xCAT) to actually provision the machine. In order to provide this, you must add an entry to the **extensions** field that contains provisioning manager-specific information. Each key in the extensions field corresponds to the provisioning manager, and certain properties are required based on this key. For example, the xCAT extension key must be named `xcat` and must contain certain fields. These extension keys are documented in [Fields: Images on page 1903](#). See the following examples of creating images with xCAT-specific provisioning information below.

#### Sample response

If the request was successful, the response body is the new image that was created exactly as shown in [Get single image](#). On failure, the response is an error message.

#### Samples

The **virtualizedImages** field only accepts input when the image is a hypervisor and expects an array of image IDs *or* names, as shown in the following example:

```
Example payload of hypervisor with 2 vms
```

```
{
  "hypervisor": true,
  "name": "esx5-stateful",
  "osType": "linux",
  "supportsPhysicalMachine": true,
  "type": "stateful",
  "hypervisorType": "ESX",
  "virtualizedImages": [
    {"id": "4fa197e68ca30fc605dd1cf0"},
    {"name": "centos5-stateful"}
  ]
}
```

The following example shows how to create an image that utilizes a cloned template for a virtual machine. (Note that the **type** must be set to `linkedclone` in order to set the **templateName** field.)

## VM Utilizing a Cloned Template

```
{
  "active": true,
  "hypervisor": false,
  "name": "centos5-compute-stateful",
  "osType": "linux",
  "type": "linkedclone",
  "supportsVirtualMachine": true,
  "templateName": "centos5-compute"
}
```

The following are samples of a virtual machine and a hypervisor image that can be provisioned with xCAT:

## xCAT Virtual Machine Image

```
{
  "active": true,
  "features": [],
  "hypervisor": false,
  "name": "centos5-compute-stateful",
  "osType": "linux",
  "type": "stateful",
  "supportsVirtualMachine": true,
  "extensions": {
    "xcat": {
      "os": "centos",
      "architecture": "x86_64",
      "profile": "compute"
    }
  }
}
```

```
xCAT Hypervisor Image
-----
{
  "active": true,
  "features": [],
  "hypervisor": true,
  "name": "esxi5-base-stateless",
  "osType": "linux",
  "virtualizedImages": [
    {"name": "centos5-compute-stateless"}
  ],
  "type": "stateless",
  "hypervisorType": "ESX",
  "supportsPhysicalMachine": true,
  "extensions": {
    "xcat": {
      "os": "esxi5",
      "architecture": "x86_64",
      "profile": "base",
      "hvType": "esx",
      "hvGroupName": "esx5hv",
      "vmGroupName": "esx5vm"
    }
  }
}
```

## Modifying images

The HTTP PUT method is used to modify **Images**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/images/<id>?api-version=3
PUT http://localhost:8080/mws/rest/images/<name>?api-version=3
```

## Modify single image

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/images/<id>?api-version=3
PUT http://localhost:8080/mws/rest/images/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Image.
<b>name</b>	Yes	String	--	The name of the Image.

**i** You must specify either **id** or **name**, but you do not have to specify both. The **name** field must contain only letters, digits, periods, dashes, and underscores.

See [Global URL parameters on page 1425](#) for available URL parameters.

Example request

```
PUT http://localhost/mws/rest/images/centos5-stateful?api-version=3
{
  "name": "centos5-stateful",
  "type": "stateful",
  "hypervisor": false,
  "osType": "linux",
  "virtualizedImages": []
}
```

**i** The **version** field contains the current version of the database entry and does not reflect the version of the operating system. This field cannot be updated directly. However, if **version** is included in the modify request, it will be used to verify that another client did not update the object in between the time the data was retrieved and the modify request was delivered.

Sample response

If the request was successful, the response body is the modified image as shown in [Get single image](#). On failure, the response is an error message.

## Deleting images

The HTTP DELETE method is used to delete **Images**.

Quick reference

```
DELETE http://localhost:8080/mws/rest/images/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/images/<name>?api-version=3
```

### Delete single image

URLs and parameters

```
DELETE http://localhost:8080/mws/rest/images/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/images/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Image.
<b>name</b>	Yes	String	--	The name of the Image.

**i** Only one of **id** or **name** are required.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON Response

```
{}
```

### Related topics

- [Fields: Images on page 1903](#)
- [Resources introduction on page 1446](#)

## Job arrays

This section describes behavior of the **Job Array** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.



The [Fields: Job Arrays](#) reference section contains the type and description of all fields in the **Job Array** object.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/job-arrays</a>	--	--	<a href="#">Submit job array</a>	--

This topic contains these sections:

- [Submitting job arrays on page 1545](#)
  - [Submit job array on page 1546](#)

## Submitting job arrays

The HTTP POST method is used to submit **Job Arrays**.

### Quick reference

```
POST http://localhost:8080/mws/rest/job-arrays?api-version=3[&proxy-user=<username>]
```



While the **Job Array** resource only gives access to create job arrays, job arrays are retrieved using the operations in [Getting job information on page 1548](#).

### Restrictions

All restrictions present for [Submitting jobs](#) are present for job arrays. In addition, job arrays are *only* supported if the `ENABLEJOBARRAYS` parameter is set to `TRUE` in the `moab.cfg` file. For example:

```
ENABLEJOBARRAYS      TRUE
```

## Submit job array

### URLs and parameters

```
POST http://localhost:8080/mws/rest/job-arrays?api-version=3 [&proxy-user=<username>]
```

Parameter	Required	Type	Valid values	Description
<b>proxy-user</b>	No	String	--	Perform this action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

To submit a job array, only two fields are required: `jobPrototype` and one of `indexValues` or `indexRanges`. Both index ranges and values may be specified if desired.

The request body below shows all the fields that are available during job array submission, although the `jobPrototype` shown is a simple example and does not utilize all fields of a job submission.

**i** The `jobPrototype` field has the same properties as a typical job submission. Consequently the `api-version` of the job array will apply to the `jobPrototype` like it does when you submit jobs, so the `api-version` in the call must match the `api-version` of the job. Examples of this can be seen in [Submitting jobs on page 1555](#).

JSON request body

```
{
  "name": "myarray",
  "indexRanges": [ {
    "startIndex": 11,
    "endIndex": 25,
    "increment": 2
  } ],
  "indexValues": [ 2, 4, 6, 8, 10 ],
  "slotLimit": 2,
  "cancellationPolicy": {
    "firstJob": "FAILURE",
    "anyJob": "SUCCESS"
  },
  "jobPrototype": {
    "commandFile": "/tmp/test.sh",
    "initialWorkingDirectory": "/tmp",
    "requirements": [{"taskCount": 4}]
  }
}
```

### Sample response

The response of this task is the same as submitting a job (see [Submit job on page 1556](#)).

Related topics

- [Fields: Job Arrays](#) on page 1915
- [Resources introduction](#) on page 1446
- [Jobs](#) on page 1547
- [Job templates](#) on page 1569

## Jobs

This section describes behavior of the **Job** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.



The Job API is new with *API version 2*. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3` in order to behave as documented.

For more information, see [Requesting specific API versions](#) on page 1428.



The [Fields: Jobs](#) reference contains the type and description of all fields in the **Job** object. It also contains details regarding which fields are valid during PUT and POST actions.

Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/jobs</code>	<a href="#">Get all jobs</a>	--	<a href="#">Submit job</a>	--
<code>/rest/jobs/&lt;name&gt;</code>	<a href="#">Get single job</a>	<a href="#">Modify job attributes</a>	--	<a href="#">Cancel job</a>
<code>/rest/jobs/&lt;name&gt;/&lt;modifyAction&gt;</code>	--	<a href="#">Perform actions on job</a>	--	--

This topic contains these sections:

- [Getting job information](#) on page 1548
  - [Get all jobs](#) on page 1548
  - [Get single job](#) on page 1549
- [Submitting jobs](#) on page 1555
  - [Submit job](#) on page 1556

- [Modifying jobs on page 1562](#)
  - [Modify job attributes on page 1562](#)
  - [Generic resources on page 1565](#)
  - [Perform actions on job on page 1567](#)
- [Deleting \(canceling\) jobs on page 1568](#)
  - [Cancel job on page 1568](#)

## Getting job information

The HTTP GET method is used to retrieve **Job** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/jobs/<name>?api-version=3
```

### Get all jobs

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/jobs?api-version=3
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"isActive":true}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

### How to get all jobs

```
GET http://localhost:8080/mws/rest/jobs?api-version=3&fields=name,flags&max=3
```

```
{
  "totalCount": 8,
  "resultCount": 3,
  "results": [
    {
      "flags": ["GLOBALQUEUE"],
      "name": "Moab.1"
    },
    {
      "flags": ["GLOBALQUEUE"],
      "name": "Moab.2"
    },
    {
      "flags": ["GLOBALQUEUE"],
      "name": "Moab.4"
    }
  ]
}
```

### How to get a subset of jobs

Get active jobs

```
http://localhost:8080/mws/rest/jobs?api-version=3&query={"isActive":true}
```

Get completed jobs

```
http://localhost:8080/mws/rest/jobs?api-version=3&query={"isActive":false}
```

Get jobs owned by a particular user

```
http://localhost:8080/mws/rest/jobs?api-version=3&query={"credentials.user":"fred"}
```

### Known issues

Some jobs are not returned if `DisplayFlags UseBlocking` is set in the `moab.cfg` file.

## Get single job

### URLs and parameters

```
GET http://localhost:8080/mws/rest/jobs/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

**i** The `attributes` field is only applicable in API version 2 and later, and the `MOAB_TENANT` field only applies if the job is attached to a tenant.

Sample response

## JSON response

```

{
  "arrayIndex": null,
  "arrayMasterName": null,
  "attributes": [],
  "blocks": [ {
    "category": "jobBlock",
    "message": null,
    "type": null
  } ],
  "bypassCount": 0,
  "cancelCount": 0,
  "commandFile": "/tmp/test.sh",
  "commandLineArgs": null,
  "completionCode": null,
  "cpuTime": 0,
  "credentials": {
    "account": null,
    "group": "adaptive",
    "jobClass": null,
    "qos": "NONE",
    "qosRequested": null,
    "user": "adaptive"
  },
  "customName": null,
  "dates": {
    "completedDate": null,
    "createdDate": "2012-10-11 17:58:16 UTC",
    "deadlineDate": "2037-10-24 12:26:40 UTC",
    "dispatchedDate": null,
    "earliestRequestedStartDate": null,
    "earliestStartDate": "2012-10-11 17:58:18 UTC",
    "eligibleDate": "2012-10-11 17:59:19 UTC",
    "lastCanceledDate": null,
    "lastChargedDate": null,
    "lastPreemptedDate": null,
    "lastUpdatedDate": "2012-10-11 17:59:19 UTC",
    "startDate": null,
    "submitDate": "2012-10-11 17:58:16 UTC",
    "terminationDate": "2037-10-24 12:26:40 UTC"
  },
  "deferCount": 0,
  "dependencies": [],
  "description": null,
  "duration": 8639999,
  "durationActive": 0,
  "durationQueued": 31,
  "durationRemaining": 0,
  "durationSuspended": 0,
  "emailNotifyAddresses": [],
  "emailNotifyTypes": [],
  "environmentRequested": false,
  "environmentVariables": {},
  "epilogScript": null,
  "flags": ["GLOBALQUEUE"],
  "holdDate": null,
  "holdReason": null,
  "holds": [],
  "initialWorkingDirectory": "/tmp",
  "isActive": true,

```

```

"jobGroup": null,
"masterNode": null,
"memorySecondsDedicated": 0,
"memorySecondsUtilized": 0,
"messages": [],
"migrateCount": 0,
"minimumPreemptTime": 0,
"mwmName": "Moab",
"name": "Moab.15",
"nodesExcluded": [],
"nodesRequested": [],
"nodesRequestedPolicy": null,
"partitionAccessList": [
  "msm",
  "SHARED"
],
"partitionAccessListRequested": [
  "msm",
  "SHARED"
],
"preemptCount": 0,
"priorities": {
  "run": 0,
  "start": 1,
  "system": 0,
  "user": 0
},
"processorSecondsDedicated": 0,
"processorSecondsLimit": 0,
"processorSecondsUtilized": 0,
"prologScript": null,
"queueStatus": "blocked",
"rejectPolicies": [],
"requirements": [ {
  "architecture": null,
  "attributes": {
    "matlab": [
      {
        "restriction": "must",
        "comparator": "<=",
        "value": "7.1",
        "displayValue": null
      }
    ]
  }
},
  {
    "MOAB_TENANT": [ {
      "value": "1234567890aabbccddeeff00",
      "displayValue": "ResearchGroup"
    }
  ]
},
  {
    "soffice": [
      {
        "restriction": "must",
        "comparator": "%=",
        "value": "3.1",
        "displayValue": null
      }
    ]
  }
],
"features": [],
"index": 0,
"featuresRequested": [],
"featuresRequestedMode": "AND",
"featuresExcluded": [],

```

```

"featuresExcludedMode": "AND",
"metrics": {},
"nodeAccessPolicy": null,
"nodeAllocationPolicy": null,
"nodeCount": 0,
"nodes": [],
"nodeSet": null,
"image": null,
"reservation": null,
"resourcesPerTask": {
  "processors": {
    "dedicated": 1,
    "utilized": 0
  },
  "memory": {
    "dedicated": 0,
    "utilized": 0
  },
  "disk": {
    "dedicated": 0,
    "utilized": null
  },
  "swap": {
    "dedicated": 0,
    "utilized": null
  }
},
"taskCount": 4,
"tasksPerNode": 0
}],
"reservationRequested": null,
"resourceFailPolicy": null,
"resourceManagerExtension": null,
"resourceManagers": [ {
  "isDestination": false,
  "isSource": true,
  "jobName": "Moab.15",
  "name": "internal"
}],
"rmStandardErrorFilePath": null,
"rmStandardOutputFilePath": null,
"standardErrorFilePath": null,
"standardOutputFilePath": null,
"startCount": 0,
"states": {
  "state": "Idle",
  "stateExpected": "Idle",
  "stateLastUpdatedDate": null,
  "subState": null
},
"submitHost": "0:0:0:0:0:0:0:1",
"systemJobAction": null,
"systemJobType": null,
"targetedJobAction": null,
"targetedJobName": null,
"templates": [{"name": "DEFAULT"}],
"triggers": [],
"variables": {},
"virtualContainers": [],
"virtualMachines": [],
"vmUsagePolicy": null
}

```

### Job arrays

- If a job is the master of a job array, the response will have some additional fields set as shown in the following example. The **name** field is chosen by the Moab, and the **customName** field comes from the [Fields: Job Arrays](#) name field.

```

Job array master
-----
{
  "name": "Moab.5",
  "customName": "myarray",
  "flags": [
    "ARRAYMASTER",
    "GLOBALQUEUE",
    "CANCELONFIRSTFAILURE",
    "CANCELONANYSUCCESS"
  ]
}

```

- If a job is a sub-job of an array, the response will have other fields set as shown in the following example.

```

Array sub-job
-----
{
  "name": "Moab.5[21]",
  "customName": "myarray",
  "arrayIndex": 21,
  "arrayMasterName": "Moab.5",
  "flags": [
    "ARRAYJOB",
    "GLOBALQUEUE",
    "CANCELONFIRSTFAILURE",
    "CANCELONANYSUCCESS"
  ]
}

```

## Submitting jobs

The HTTP POST method is used to submit **Jobs**.

### Quick reference

```
POST http://localhost:8080/mws/rest/jobs?api-version=3[&proxy-user=<username>]
```

### Restrictions

- No more than one virtual container can be specified in the request. The virtual container must already exist.
- The `credentials.user` and `credentials.group` properties are used to submit a job as the specified user belonging to the specified group.

- Job variables have the following restrictions:
  - Variable names cannot contain equals (=), semicolon (;), colon (:), plus (+), question mark (?), caret (^), backslash (\), or white space.
  - Variable values cannot contain semicolon (;), colon (:), plus (+), or caret (^).
- When submitting jobs, the only supported hold type is `User`.
- The proxy-user parameter is ignored unless you set `ENABLEPROXY=TRUE` in the `moab.cfg` file. For example:

```
ADMINCFG[1]          USERS=root,ted ENABLEPROXY=TRUE
```

## Submit job

### URLs and parameters

```
POST http://localhost:8080/mws/rest/jobs?api-version=3 [&proxy-user=<username>]
```

Parameter	Required	Type	Valid values	Description
<b>proxy-user</b>	No	String	--	Perform the action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

JSON request body (specified host list)

```
{
  "attributes": [
    "attr1",
    "attr2"
  ],
  "commandFile": "/tmp/test.sh",
  "commandScript": "c2xlZXAgNjAK",
  "commandLineArguments": "-x -v",
  "credentials": {
    "account": "account",
    "group": "group",
    "jobClass": "BATCH",
    "qosRequested": "QOS1",
    "user": "saadmin"
  },
  "customName": "custom name for job",
  "dates": {
    "earliestRequestedStartDate": "2012-11-08 13:18:47 UTC",
    "deadlineDate": "2014-02-17 14:00:00 UTC"
  },
  "dependencies": [
    {
      "type": "set",
      "name": "vc1.varA"
    },
    {
      "type": "set",
      "name": "vc2.varB"
    },
    {
      "type": "set",
      "name": "vc3.varC"
    }
  ],
  "duration": 600,
  "emailNotifyAddresses": [
    "user3@ac.com",
    "user4@ac.com"
  ],
  "emailNotifyTypes": [
    "JobStart",
    "JobEnd"
  ],
  "environmentRequested": true,
  "environmentVariables": {
    "var1": "val1",
    "var2": "val2"
  },
  "epilogScript": "/tmp/epilog.sh",
  "flags": [
    "RESTARTABLE",
    "SUSPENDABLE"
  ],
  "holds": ["User"],
  "initialWorkingDirectory": "/tmp",
  "jobGroup": "job_group",
  "nodesExcluded": [
    {"name": "node07"},
    {"name": "node08"}
  ]
}
```

```

],
"nodesRequested": [
  {"name": "node01"},
  {"name": "node02"}
],
"nodesRequestedPolicy": "SUBSET",
"partitionAccessListRequested": [
  "p1",
  "p2"
],
"priorities": {"user": 5},
"prologScript": "/tmp/prolog.sh",
"requirements": [ {
  "architecture": "x86_64",
  "attributes": {
    "matlab": [
      {
        "restriction": "must",
        "comparator": "<=",
        "value": "7.1"
      }
    ]
  },
  "soffice": [
    {
      "restriction": "must",
      "comparator": "%=",
      "value": "3.1"
    }
  ]
}
],
"featuresRequested": [
  "a",
  "b",
  "c"
],
"featuresRequestedMode": "OR",
"featuresExcluded": [
  "d",
  "e",
  "f"
],
"featuresExcludedMode": "AND",
"nodeAccessPolicy": "SINGLEJOB",
"nodeAllocationPolicy": "PRIORITY",
"nodeCount": 6,
"nodeSet": "FIRSTOF:FEATURE:vlan2",
"image": "linux",
"resourcesPerTask": {
  "disk": {"dedicated": 1024},
  "memory": {"dedicated": 512},
  "processors": {"dedicated": 2},
  "swap": {"dedicated": 4096},
  "matlab": {"dedicated": 6},
  "intellij": {"dedicated": 2}
},
"taskCount": 4,
"tasksPerNode": 14
}],
"reservationRequested": {"name": "rsv.1"},
"resourceFailPolicy": "RETRY",
"resourceManagerExtension": "x=PROC=4",
"standardErrorFilePath": "/tmp/error",

```

```

"standardOutputFilePath": "/tmp/out",
"submitHost": "admin-node",
"templates": [
  {"name": "template1"},
  {"name": "template2"}
],
"variables": {
  "var1": "val1",
  "var2": "val2"
},
"virtualContainers": [{"name": "vc1"}],
"vmUsagePolicy": "CREATEVM"
}

```

### Sample response

The response of this task is one of three possibilities:

- An object with a single **messages** property containing a list of error messages on failure.

```

{"messages":["Could not create job - invalid requirements"]}

```

- An object with a **name** property containing the name of the newly created job.

```

{"name":"Moab.1"}

```

- An object with a **name** property and a **virtualContainers** list containing the name of the newly created virtual container.

```

{ "name": "Moab.1", "virtualContainers": [{"name": "vc1"}] }

```



The virtual container will only be reported when a *new* virtual container has been created by Moab for the job.

### Examples of job submission

This section includes some sample job submission requests.

#### Example 4-5: Submit job to run on node2 and node3

```

POST http://localhost:8080/mws/rest/jobs?api-version=3
-----
{
  "commandFile": "/tmp/test.sh",
  "credentials": {
    "group": "adaptive",
    "user": "adaptive"
  },
  "initialWorkingDirectory": "/tmp",
  "nodesRequested": [
    {"name": "node2"},
    {"name": "node3"}
  ]
}

```

**Example 4-6: Submit job that requires 20 processors**

```
POST http://localhost:8080/mws/rest/jobs?api-version=3
-----
{
  "commandFile": "/tmp/test.sh",
  "credentials": {
    "group": "adaptive",
    "user": "adaptive"
  },
  "initialWorkingDirectory": "/tmp",
  "requirements": [{"taskCount": 20}]
}
```

**Example 4-7: Submit job to run after a certain time**

```
POST http://localhost:8080/mws/rest/jobs?api-version=3
-----
{
  "commandFile": "/tmp/test.sh",
  "credentials": {
    "group": "adaptive",
    "user": "adaptive"
  },
  "dates": {"earliestRequestedStartDate": "2012-10-11 18:36:35 UTC"},
  "initialWorkingDirectory": "/tmp",
  "requirements": [{"taskCount": 20}]
}
```

**Example 4-8: Submit job based on *msub* example**

Given this *msub* command:

```
msub -l nodes=3:ppn=2,walltime=1:00:00,pmem=100 script2.pbs.cmd
```

Here is an equivalent MWS request:

```
POST http://localhost:8080/mws/rest/jobs?api-version=3
-----
{
  "duration": 3600,
  "commandFile": "/home/adaptive/script2.pbs.cmd",
  "credentials": {
    "group": "adaptive",
    "user": "adaptive"
  },
  "initialWorkingDirectory": "/home/adaptive",
  "requirements": [ {
    "resourcesPerTask": {"memory": {"dedicated": 100}},
    "taskCount": 6,
    "tasksPerNode": 2
  } ]
}
```

**i** To emulate what `msub` does, make `commandFile` an absolute path, and add `credentials.user`, `credentials.group`, and `initialWorkingDirectory`.

As shown above, `nodes=3:ppn=2` is equivalent to setting `taskCount` to 6 and `tasksPerNode` to 2.

Example 4-9: Submit a job array

For information on how to submit a job array, see [Submitting job arrays on page 1545](#).

## Modifying jobs

The HTTP PUT method is used to modify **Jobs**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/jobs/<name>[/<modifyAction>]?api-version=3 [&proxy-user=<username>]
```

### Restrictions

The **proxy-user** parameter is ignored unless you set `ENABLEPROXY=TRUE` in the `moab.cfg` file. For example:

```
ADMINCFG[1]          USERS=root,ted ENABLEPROXY=TRUE
```

## Modify job attributes

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/jobs/<name>?api-version=3 [&proxy-user=<username>] [&change-mode=set]
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.
<b>proxy-user</b>	No	String	--	Perform the action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Additional URL parameters

Parameter	Required	Valid values	Description
<b>change-mode</b>	No	set (default) add remove	If <b>set</b> , replace all fields with the fields specified. If <b>add</b> , add the specified fields to existing fields. If <b>remove</b> , remove the specified fields from existing fields.

Request body

The request body below shows all the fields that are available when modifying a job, along with some sample values.

JSON request body

```
{
  "credentials": {
    "account": "account",
    "jobClass": "BATCH",
    "qosRequested": "QOS1"
  },
  "customName": "custom name for job",
  "dates": {"earliestRequestedStartDate": "2012-11-08 13:18:47 UTC"},
  "duration": 600,
  "flags": [
    "RESTARTABLE",
    "SUSPENDABLE"
  ],
  "holds": ["User"],
  "messages": [
    {"message": "Message one"},
    {"message": "Message two"}
  ],
  "nodesRequested": [
    {
      "name": "n015"},
    {
      "name": "n016"
    },
    {
      "name": "n017"
    },
    {
      "name": "n018"
    }
  ],
  "partitionAccessListRequested": [
    "p1",
    "p2"
  ],
  "priorities": {
    "system": 3,
    "user": 5
  },
  "requirements": [
    {
      "features": [
        "vlan1",
        "vlan2"
      ],
      "resourcesPerTask": [
        "matlab": {
          "dedicated": 1
        },
        "tape": {
          "dedicated": 2
        }
      ]
    }
  ],
  "reservationRequested": {"name": "rsv.1"},
  "variables": {
    "var1": "val1",
    "var2": "val2"
  }
}
```

```
}
}
```

### Sample response

**i** These messages may not match the messages returned from Moab exactly, but are given as an example of the structure of the response.

**i** Not all messages are shown for the above request body.

JSON response

```
-----
{"messages": [
  "Account modified successfully",
  "Messages modified successfully",
  "Variables modified successfully"
]}
```

### Restrictions

- Old messages are not removed from jobs; only new messages are added.
- Job variables have the restrictions documented in [Submitting jobs on page 1555](#).
- Although the client can modify `features` and `resourcesPerTask`, Moab only considers these elements when they appear in the first element of the `requirements` array. If the `requirements` array contains two or more elements, all elements but the first are silently ignored.

### Generic resources

Jobs can require configurable, site-specific consumable resources called generic resources. For example, some jobs may require a matlab license. Only one job at a time may legally consume this license. Matlab is not a standard resource and may only be available on some sites. Nevertheless Moab allows this to be configured and tracked as is explained in [Managing Consumable Generic Resources on page 573](#).

You must specify generic resources in the `requirements.resourcesPerTask` portion of the JSON document. Any resource in `requirement.resourcesPerTask` that is not a standard resource is considered a generic resource. Standard resources include disk, memory, processors, and swap. Assume a job has the following in `requirement.resourcesPerTask`:

```
{
  "resourcesPerTask":{
    "processors":{
      "dedicated":4,
      "utilized":0
    },
    "memory":{
      "dedicated":2048,
      "utilized":0
    },
    "disk":{
      "dedicated":4096,
      "utilized":0
    },
    "swap":{
      "dedicated":1024,
      "utilized":0
    },
    "tape":{
      "dedicated":1,
      "utilized":0
    },
    "matlab":{
      "dedicated":2,
      "utilized":0
    }
  }
}
```

The standard resources the job requires are:

- 4 processors
- 2048 MB of memory
- 4096 MB of disk
- 1024 MB of swap

The generic resources the job requires are

- 1 tape
- 2 matlab

To modify a job so that it requires 1 matlab license, run the following:

```
PUT http://localhost:8080/mws/rest/jobs/Moab.2?api-version=3
{
  "requirements":[
    {
      "resourcesPerTask":{
        "matlab":{
          "dedicated":1
        }
      }
    }
  ]
}
```

## Perform actions on job

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/jobs/<name>/<modifyAction>?api-version=3 [&proxy-user=<username>]
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.
<b>modifyAction</b>	Yes	String	cancel checkpoint execute hold requeue rerun resume suspend unhold	<p>If <b>cancel</b>, attempts to cancel the job (equivalent to deleting a job).</p> <p>If <b>checkpoint</b>, attempts to checkpoint the job. Note that the OS must support checkpointing for this to work.</p> <p>If <b>execute</b>, executes the job (if possible).</p> <p>If <b>hold</b>, attempts to hold the job using the holds set in the request body.</p> <p>If <b>requeue</b>, attempts to requeue the job.</p> <p>If <b>rerun</b>, attempts to rerun the job.</p> <p>If <b>resume</b>, attempts to resume the job.</p> <p>If <b>suspend</b>, attempts to suspend the job.</p> <p>If <b>unhold</b>, attempts to release the holds set in the request body.</p>
<b>proxy-user</b>	No	String	--	Perform the action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

Request bodies are only required for holding or unholding jobs. All other actions do not require request bodies of any kind.

JSON request body to add holds to a job

```
{"holds": ["User"]}
```

JSON request body to remove holds from a job

```
{"holds": ["User"]}
```



If no holds are specified when unholding a job, all holds will be removed. This is equivalent to specifying holds as a list with a single element of `All`.

Sample response

**i** This message may not match the message returned from Moab exactly, but is given as an example of the structure of the response.

JSON response

```
-----
{"messages": ["Job modified successfully"]}
```

## Deleting (canceling) jobs

The HTTP DELETE method is used to cancel **Jobs**.

Quick reference

```
DELETE http://localhost:8080/mws/rest/jobs/<name>?api-version=3[&proxy-
user=<username>]
```

Restrictions

The **proxy-user** parameter is ignored unless you set `ENABLEPROXY=TRUE` in the `moab.cfg` file. For example:

```
ADMINCFG[1]          USERS=root,ted ENABLEPROXY=TRUE
```

## Cancel job

URLs and parameters

```
DELETE http://localhost:8080/mws/rest/jobs/<name>?api-version=3[&proxy-
user=<username>]
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.
<b>proxy-user</b>	No	String	--	Perform the action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

JSON response for successful DELETE

```
-----
{}
```

**i** Additional information about the DELETE can be found in the HTTP response header X-MWS-Message.

Related topics

- [Fields: Jobs](#) on page 2006
- [Resources introduction](#) on page 1446
- [Job arrays](#) on page 1545
- [Job templates](#) on page 1569

## Job templates

This section describes behavior of the **Job Template** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Job Templates](#) reference section contains the type and description of all fields in the **Job Template** object. It also contains details regarding which fields are valid during PUT and POST actions.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/job-templates</a>	<a href="#">Get all job templates</a>	--	--	--
<a href="#">/rest/job-templates/&lt;id&gt;</a>	<a href="#">Get single job template</a>	--	--	--

This topic contains these sections:

- [Getting job templates](#) on page 1569
  - [Get all job templates](#) on page 1570
  - [Get single job template](#) on page 1570

### Getting job templates

The HTTP GET method is used to retrieve **Job Template** information. Queries for all objects and a single object are available.

Quick reference

```
GET http://localhost:8080/mws/rest/job-templates/<id>?api-version=3
```

## Get all job templates

### URLs and parameters

```
GET http://localhost:8080/mws/rest/job-templates?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/job-templates?api-version=3&fields=id
{
  "totalCount": 14,
  "resultCount": 14,
  "results": [
    {"id": "DEFAULT"},
    {"id": "genericVM"},
    {"id": "genericVM-setup"},
    {"id": "genericVM-destroy"},
    {"id": "genericVM-migrate"},
    {"id": "genericPM"},
    {"id": "genericPM-setup"},
    {"id": "genericPM-destroy"},
    {"id": "OSStorage"},
    {"id": "OSStorage-setup"},
    {"id": "OSStorage-destroy"},
    {"id": "extraStorage"},
    {"id": "extraStorage-setup"},
    {"id": "extraStorage-destroy"}
  ]
}
```

## Get single job template

### URLs and parameters

```
GET http://localhost:8080/mws/rest/job-templates/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
id	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

## Sample response

JSON response

```

{
  "account": "account",
  "args": "arg1 arg2",
  "commandFile": "/tmp/script",
  "description": "description",
  "genericSystemJob": true,
  "id": "genericVM",
  "inheritResources": false,
  "jobDependencies": [ {
    "name": "genericVM-setup",
    "type": "JOBSUCCESSFULCOMPLETE"
  } ],
  "jobFlags": ["VMTRACKING"],
  "jobTemplateFlags": ["SELECT"],
  "jobTemplateRequirements": [ {
    "architecture": "x86_64",
    "diskRequirement": 500,
    "genericResources": {"tape": 3},
    "nodeAccessPolicy": "SINGLEJOB",
    "operatingSystem": "Ubuntu 10.04.3",
    "requiredDiskPerTask": 200,
    "requiredFeatures": ["dvd"],
    "requiredMemoryPerTask": 1024,
    "requiredProcessorsPerTask": 2,
    "requiredSwapPerTask": 512,
    "taskCount": 4
  } ],
  "priority": 20,
  "qos": "qos",
  "queue": "queue",
  "durationRequested": 600,
  "select": true,
  "trigger": null,
  "version": 0,
  "vmUsagePolicy": "REQUIREPM"
}

```

## Related topics

- [Fields: Job Templates on page 2107](#)
- [Resources introduction on page 1446](#)
- [Jobs on page 1547](#)
- [Job arrays on page 1545](#)

## Metric types

This section describes behavior of the **Metric Type** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.



The [Fields: Metric Types](#) reference section contains the type and description of all fields in the **Metric Type** object.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/metric-types</a>	<a href="#">Get all metric types</a>	--	--	--

This topic contains these sections:

- [Getting metric types on page 1572](#)
  - [Get all metric types on page 1572](#)

## Getting metric types

The HTTP GET method is used to retrieve **Metric Type** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/metric-types?api-version=3
```

### Get all metric types

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/metric-types?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/metric-types?api-version=3&fields=id
```

```
{
  "totalCount": 9,
  "resultCount": 9,
  "results": [
    {"id": "vmcount"},
    {"id": "watts"},
    {"id": "pwatts"},
    {"id": "temp"},
    {"id": "cpu"},
    {"id": "mem"},
    {"id": "io"},
    {"id": "ccores"},
    {"id": "threads"}
  ]
}
```

#### Related topics

- [Fields: Metric Types on page 2149](#)
- [Resources introduction on page 1446](#)

## Nodes

This section describes behavior of the **Node** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.



The Node API is new with *API version 2*. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3` in order to behave as documented.

For more information, see [Requesting specific API versions on page 1428](#).



The **Fields: Nodes** reference contains the type and description of all fields in the **Node** object. It also contains details regarding which fields are valid during PUT and POST actions.

### Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/nodes</code>	<a href="#">Get all nodes</a>	--	--	--
<code>/rest/nodes/&lt;name&gt;</code>	<a href="#">Get single node</a>	<a href="#">Modify node</a>	--	--

This topic contains these sections:

- [Getting nodes on page 1573](#)
  - [Get all nodes on page 1573](#)
  - [Get single node on page 1574](#)
- [Modifying nodes on page 1578](#)
  - [Modify node on page 1579](#)

## Getting nodes

The HTTP GET method is used to retrieve **Node** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/nodes/<name>?api-version=3
```

### Get all nodes

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/nodes?api-version=3
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"type":"-compute"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

**i** This query will not return the `DEFAULT` or `GLOBAL` nodes from Moab. However, the [Get single node](#) task may be used to retrieve them individually if desired.

Sample response

```
GET http://localhost:8080/mws/rest/nodes?api-version=3&fields=name
-----
{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {"name": "node1"},
    {"name": "node2"},
    {"name": "node3"}
  ]
}
```

**Get single node**

URLs and parameters

```
GET http://localhost:8080/mws/rest/nodes/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

**i** The `attributes` field is only applicable in API version 2 and later, and the `MOAB_TENANT` field only applies if the node is attached to a tenant.

Sample response

JSON response

```
{
  "name": "126.csa",
  "architecture": null,
  "classes": ["class1"],
  "attributes": {
    "MOAB_TENANT": {
      "value": "1234567890abcdef12345678",
      "displayValue": "ResearchGroup"
    },
    "MOAB_DATACENTER": {
      "value": "vcenter-datacenter-401",
      "displayValue": "vcenter-vcenter - adaptive data center"
    },
    "vcenter-vcenter-adaptive data center-compute nodes": {
      "value": null,
      "displayValue": null
    }
  },
  "featuresCustom": ["feature1", "feature2"],
}
```

```

"featuresReported": ["vcenter-vcenter-adaptive data center-compute nodes"],
"index": 26,
"ipAddress": "10.0.8.76",
"isHypervisor": true,
"lastUpdatedDate": "2013-05-24 20:18:11 UTC",
"migrationDisabled": false,
"partition": "mws",
"processorSpeed": null,
"profilingEnabled": false,
"rack": null,
"resourceManagerMessages": {
  "torque": null,
  "mws": null
},
"slot": null,
"type": "compute",
"messages": [ {
  "count": 11,
  "createdDate": "2012-10-24 04:06:04 UTC",
  "expireDate": "2037-10-24 12:26:40 UTC",
  "message": "This is a message"
}],
"metrics": {
  "vmcount": 0,
  "cpuUtilization": 0.275,
  "cpuLoad": 0.01115
},
"variables": {
  "VCENTER_DATASTORE_LOCAL1": "datastore-415",
  "VCENTER_DATASTORE_REMOTE1": "datastore-448"
},
"states": {
  "powerState": "On",
  "powerStateExpected": null,
  "state": "Idle",
  "stateExpected": "Idle",
  "stateLastUpdatedDate": "2013-05-24 09:33:45 UTC",
  "subState": null,
  "subStateLast": null,
  "subStateLastUpdatedDate": null
},
"operatingSystem": {
  "hypervisorType": "esx",
  "image": "vcenter-vcenter-esx-5.0",
  "imageExpected": null,
  "imageLastUpdatedDate": null,
  "imagesAvailable": [],
  "virtualMachineImages": [
    "win2008",
    "centos6"
  ]
},
"resources": {
  "processors": {
    "configured": 4,
    "real": 4,
    "dedicated": 0,
    "available": 4,
    "utilized": -1
  },
  "memory": {
    "configured": 10239,

```

```

    "real": 10239,
    "dedicated": 0,
    "available": 9227,
    "utilized": 0
  },
  "disk": {
    "configured": 0,
    "real": 0,
    "dedicated": 0,
    "available": 0,
    "utilized": 0
  },
  "swap": {
    "configured": 0,
    "real": 0,
    "dedicated": 0,
    "available": 0,
    "utilized": 0
  }
},
"resourceManagers": [ {
  "name": "mws",
  "isMaster": true,
  "stateReported": "Active"
}],
"jobs": [],
"reservations": [
  {
    "name": "system.5",
    "type": "user"
  },
  {
    "name": "system.17",
    "type": "user"
  }
],
"virtualContainers": [],
"virtualMachines": [],
"triggers": []
}

```

## Modifying nodes

The HTTP PUT method is used to modify **Nodes**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/nodes/<name>?api-version=3 [&proxy-user=<username>]
```

### Restrictions

The **proxy-user** parameter is ignored unless you set `ENABLEPROXY=TRUE` in the `moab.cfg` file. For example:

```
ADMINCFG[1]          USERS=root,ted ENABLEPROXY=TRUE
```

## Modify node

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/nodes/<name>?api-version=3 [&proxy-user=<username>]
[&change-mode=set]
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the object.
<b>proxy-user</b>	No	String	--	Perform the action as this user.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Additional URL parameters

Parameter	Required	Valid values	Description
<b>change-mode</b>	No	set (default) add remove	If <b>set</b> , replace all features with the features specified. If <b>add</b> , add the specified features to existing features. If <b>remove</b> , remove the specified features from existing features.

### Request body

The request body below shows all the fields that are available when modifying a node, along with some sample values.

Sample JSON request body to modify a node

```
{
  "featuresCustom": ["feature1", "feature2"],
  "messages": [
    {"message": "Message one"},
    {"message": "Message two"}
  ],
  "metrics": {"pwatts": 211},
  "operatingSystem": {"image": "esx4.1"},
  "partition": "part1",
  "states": {
    "powerState": "On",
    "state": "Running"
  },
  "variables": {
    "key": "value",
    "arbitrary text key": "more value"
  }
}
```

Sample response

**i** This message may not match the message returned from Moab exactly, but is given as an example of the structure of the response.

JSON response

```

{
  "messages": [
    "Successfully modified os to 'linux'",
    "Successfully powered node off"
  ]
}
    
```

**Related topics**

- [Fields: Nodes on page 2150](#)
- [Resources introduction on page 1446](#)

## Notification conditions

This section describes behavior of the **Notification Conditions** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**!** The Notification Conditions API is new with *API version 3*, and is not available with older API versions. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3`.

For more information, see [Requesting specific API versions on page 1428](#).

**i** The [Fields: Notification Conditions](#) reference contains the type and description of all fields in the **Notification Conditions** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/notification-conditions</b>	<a href="#">Get all notification conditions</a>	<a href="#">Update notification condition</a>	--	--
<b>/rest/notification-conditions/&lt;id&gt;</b>	<a href="#">Get single notification condition</a>	--	--	--

This topic contains these sections:

- [Getting notification conditions on page 1581](#)
  - [Get all notification conditions on page 1581](#)
  - [Get single notification condition on page 1583](#)
- [Updating notification conditions on page 1583](#)
  - [Update notification condition on page 1584](#)

## Getting notification conditions

The HTTP GET method is used to retrieve **Notification Condition** information.

### [Quick reference](#)

```
GET http://localhost:8080/mws/rest/notification-conditions?api-version=3
GET http://localhost:8080/mws/rest/notification-conditions/<id>?api-version=3
```

### Get all notification conditions

#### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/notification-conditions?api-version=3[&query=
{"escalationLevel":"ADMIN"}][&sort={"observedDate":-1}]
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Query for specific results. It is possible to query notifications by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"escalationLevel":"ADMIN"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"observedDate":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```

GET http://localhost:8080/mws/rest/notification-conditions?api-version=3&query=
{"escalationLevel":"ADMIN"}&sort={"observedDate":-1}
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "createdDate": "2013-09-10 23:13:33 UTC",
      "details": {
        "pluginType": "NodeUtilizationReport",
        "pluginId": "node-report"
      },
      "escalationLevel": "ADMIN",
      "expirationDate": null,
      "expirationDuration": null,
      "message": "The node 'testnode' has not been updated since the last poll,
which is likely due to a misconfiguration.",
      "objectId": "testnode",
      "objectType": "Node",
      "observedDate": "2013-09-10 23:13:33 UTC",
      "origin": "MWS/plugins/NodeUtilizationReport/node-report",
      "tenant": {
        "id": "1234567890abcdef12345678",
        "name": "Research"
      },
      "id": "522fa79de4b0cafeaec6f83e"
    },
    {
      "createdDate": "2013-09-11 17:19:35 UTC",
      "details": {
        "pluginType": "VCenter",
        "pluginId": "vcenter42"
      },
      "escalationLevel": "ADMIN",
      "expirationDate": null,
      "expirationDuration": null,
      "message": "The node 'node1' does not have vcenter tools installed,
therefore the state is unknown and migrations may not work correctly",
      "objectId": null,
      "objectType": "System",
      "observedDate": "2013-09-11 17:19:35 UTC",
      "origin": "MWS/plugins/VCenter/vcenter42",
      "tenant": {
        "id": "1234567890abcdef12345678",
        "name": "Research"
      },
      "id": "5230a627e4b0d51bef490e86"
    }
  ]
}

```

**i** A notification's `tenant` is automatically inherited from the `objectId` and `objectType` fields. If no object is associated with the notification condition, the notification is visible to all tenants.

## Get single notification condition

### URLs and parameters

```
GET http://localhost:8080/mws/rest/notification-conditions/<id>?api-version=3
```

Parameter	Required	Type	Description
<b>id</b>	Yes	String	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/notification-conditions/521a1f18e4b0e3f9031f47f5?api-version=3
```

```
{
  "createdDate": "2013-09-10 23:13:33 UTC",
  "details": {
    "pluginType": "NodeUtilizationReport",
    "pluginId": "node-report"
  },
  "escalationLevel": "ADMIN",
  "expirationDate": null,
  "expirationDuration": null,
  "message": "The node 'testnode' has not been updated since the last poll, which is likely due to a misconfiguration.",
  "objectId": "testnode",
  "objectType": "Node",
  "observedDate": "2013-09-10 23:13:33 UTC",
  "origin": "MWS/plugins/NodeUtilizationReport/node-report",
  "tenant": {
    "id": "1234567890abcdef12345678",
    "name": "Research"
  },
  "id": "522fa79de4b0cafeaec6f83e"
}
```

**i** A notification's tenant is automatically inherited from the `objectId` and `objectType` fields. If no object is associated with the notification condition, the notification is visible to all tenants.

## Updating notification conditions

The HTTP PUT method is used to update **Notification Condition** information. The PUT operation is idempotent, meaning that is used for both creating new notification conditions and updating existing ones. If the `escalationLevel`, `origin`, `message`, `objectType`, and `objectId` fields match an existing notification condition, it will be updated. Otherwise, a new condition will be created.

### Quick reference

```
PUT http://localhost:8080/mws/rest/notification-conditions?api-version=3
```

## Update notification condition

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notification-conditions?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

The request body below shows some fields that are available when updating a notification condition, along with some sample values.

Sample JSON request body to update a notification condition

```
{
  "details": {
    "pluginType": "NodeTester",
    "pluginId": "my-tester1"
  },
  "escalationLevel": "ADMIN",
  "expirationDuration": 30,
  "message": "Node 'node2' is powered off, please check your hardware.",
  "objectId": "node2",
  "objectType": "Node",
  "origin": "NodeTester/my-tester1/Test.groovy:141"
}
```

### Sample response

JSON response

```
{
  "createdDate": "2013-09-10 23:13:33 UTC",
  "details": {
    "pluginType": "NodeTester",
    "pluginId": "my-tester1"
  },
  "escalationLevel": "ADMIN",
  "expirationDate": "2013-09-10 23:14:03 UTC",
  "expirationDuration": 30,
  "observedDate": "2013-09-10 23:13:33 UTC",
  "message": "Node 'node2' is powered off, please check your hardware.",
  "objectId": "node2",
  "objectType": "Node",
  "origin": "NodeTester/my-tester1/Test.groovy:141",
  "tenant": {
    "id": "1234567890abcdef12345678",
    "name": "Research"
  },
  "id": "5230a627e4b0d51bef490e86"
}
```

### Related topics

- [Resources introduction on page 1446](#)
- [Events on page 1528](#)

- [Fields: Events](#) on page 1894
- [Notifications](#) on page 1585
- [Fields: Notifications](#) on page 2195
- [Fields: Notification Conditions](#) on page 2191
- [Creating events and notifications](#) on page 1704
- [Plugin event service](#) on page 1755
- [Handling events](#) on page 1711
- [System events](#) on page 1444
- [Securing the connection with the message queue](#) on page 1417

## Notifications

This section describes behavior of the **Notifications** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.



The Notifications API is new with *API version 3*, and is not available with older API versions. The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3`.

For more information, see [Requesting specific API versions](#) on page 1428.



The [Fields: Notifications](#) reference contains the type and description of all fields in the **Notifications** object.

### Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/notifications/</code>	<a href="#">Get all notifications</a>	--	--	--
<code>/rest/notifications/&lt;id&gt;</code>	<a href="#">Get single notification</a>	--	--	--
<code>/rest/notifications/ignore</code>	--	<a href="#">Ignore all notifications</a>	--	--
<code>/rest/notifications/&lt;id&gt;/ignore</code>	--	<a href="#">Ignore single notification</a>	--	--
<code>/rest/notifications/unignore</code>	--	<a href="#">Unignore all notifications</a>	--	--

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/notifications/&lt;id&gt;/unignore</a>	--	<a href="#">Unignore single notification</a>	--	--
<a href="#">/rest/notifications/dismiss</a>	--	<a href="#">Dismiss all notifications</a>	--	--
<a href="#">/rest/notifications/&lt;id&gt;/dismiss</a>	--	<a href="#">Dismiss single notification</a>	--	--

This topic contains these sections:

- [Getting notifications on page 1586](#)
  - [Get all notifications on page 1586](#)
  - [Get single notification on page 1588](#)
- [Ignoring notifications on page 1589](#)
  - [Ignore all notifications on page 1589](#)
  - [Ignore single notification on page 1589](#)
- [Unignoring notifications on page 1590](#)
  - [Unignore all notifications on page 1590](#)
  - [Unignore single notification on page 1591](#)
- [Dismissing notifications on page 1591](#)
  - [Dismiss all notifications on page 1591](#)
  - [Dismiss single notification on page 1592](#)

## Getting notifications

The HTTP GET method is used to retrieve **Notification** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/notifications?api-version=3
GET http://localhost:8080/mws/rest/notifications/<id>?api-version=3
```

## Get all notifications

### URLs and parameters

```
GET http://localhost:8080/mws/rest/notifications?api-version=3[&proxy-user=<username>]
[&query={"ignoredDate":null,"dismissedDate":null}][&sort={"observedDate":-1}]
```

Parameter	Required	Type	Description	Example
<b>proxy-user</b>	No	String	Perform the action as this user. <div style="border: 1px solid #0070c0; padding: 5px; margin-top: 10px;">  Notifications cannot be created directly. Instead, they are automatically created for the current user or <code>proxy-user</code> specified in the request from non-expired notification conditions (see <a href="#">Notification conditions</a> on page 1580). This is true no matter the query specified.                     </div>	--
<b>query</b>	No	JSON	Query for specific results. It is possible to query notifications by one or more fields based on <a href="#">MongoDB query syntax</a> . However, typically you will want to query on <code>{"ignoredDate":null,"dismissedDate":null}</code> .	<pre>query= {"ignoredDate":null,"dismissedDate":null}</pre>
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	<pre>sort={"observedDate":-1}</pre>

See [Global URL parameters](#) on page 1425 for available URL parameters.

**Sample response**

```
GET http://localhost:8080/mws/rest/notifications?api-version=3&proxy-user=<username>&query={"ignoredDate":null,"dismissedDate":null} [&sort={"observedDate":-1}]
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "conditionId": "521bdeale4b019cd33e29c86",
      "createdDate": "2013-08-26 23:02:56 UTC",
      "details": {},
      "dismissedDate": null,
      "ignoredDate": null,
      "message": "A health check failed for the 'ZeroMQ Message Queue' connection, please see the MWS health details page for more information.",
      "objectId": "zmq",
      "objectType": "Health",
      "observedDate": "2013-09-05 17:57:00 UTC",
      "origin": "MWS/HealthNotificationJob",
      "user": "admin",
      "id": "5230ed82e4b065347016d62f"
    },
    {
      "conditionId": "521a1f18e4b0e3f9031f47f5",
      "createdDate": "2013-08-25 15:13:28 UTC",
      "details": {},
      "dismissedDate": null,
      "ignoredDate": null,
      "message": "A health check failed for the 'LDAP' connection, please see the MWS health details page for more information.",
      "objectId": "ldap",
      "objectType": "Health",
      "observedDate": "2013-08-30 18:11:15 UTC",
      "origin": "MWS/HealthNotificationJob",
      "user": "admin",
      "id": "5230ed82e4b065347016d60d"
    }
  ]
}
```

**Get single notification**

URLs and parameters

```
GET http://localhost:8080/mws/rest/notifications/<id>?api-version=3
```

Parameter	Required	Type	Description
<b>id</b>	Yes	String	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d?api-
version=3
-----
{
  "conditionId": "521a1f18e4b0e3f9031f47f5",
  "createdDate": "2013-08-25 15:13:28 UTC",
  "details": {},
  "dismissedDate": null,
  "ignoredDate": null,
  "message": "A health check failed for the 'LDAP' connection, please see the MWS
health details page for more information.",
  "objectId": "ldap",
  "objectType": "Health",
  "observedDate": "2013-08-30 18:11:15 UTC",
  "origin": "MWS/HealthNotificationJob",
  "user": "admin",
  "id": "5230ed82e4b065347016d60d"
}
```

## Ignoring notifications

The HTTP PUT method is used to ignore **Notifications**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/notifications/ignore?api-version=3
PUT http://localhost:8080/mws/rest/notifications/<id>/ignore?api-version=3
```

## Ignore all notifications

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notifications/ignore?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
JSON response
-----
{"messages":["Updated 10 Notification objects"]}
```

## Ignore single notification

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/ignore?api-
version=3
```

Parameter	Required	Type	Description
<b>id</b>	Yes	String	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
PUT http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/ignore?api-version=3
-----
{
  "conditionId": "521a1f18e4b0e3f9031f47f5",
  "createdDate": "2013-08-25 15:13:28 UTC",
  "details": {},
  "dismissedDate": null,
  "ignoredDate": "2013-09-17 15:34:36 UTC",
  "message": "A health check failed for the 'LDAP' connection, please see the MWS health details page for more information.",
  "objectId": "ldap",
  "objectType": "Health",
  "observedDate": "2013-08-30 18:11:15 UTC",
  "origin": "MWS/HealthNotificationJob",
  "user": "admin",
  "id": "5230ed82e4b065347016d60d"
}
```

## Unignoring notifications

The HTTP PUT method is used to unignore **Notifications**.

#### Quick reference

```
PUT http://localhost:8080/mws/rest/notifications/unignore?api-version=3
PUT http://localhost:8080/mws/rest/notifications/<id>/unignore?api-version=3
```

### Unignore all notifications

#### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notifications/unignore?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
JSON response
-----
{"messages":["Updated 10 Notification objects"]}
```

## Unignore single notification

```
PUT
http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/unignore?api-
version=3
```

Parameter	Required	Type	Description
<b>id</b>	Yes	String	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
PUT
http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/unignore?api-
version=3
```

```
{
  "conditionId": "521a1f18e4b0e3f9031f47f5",
  "createdDate": "2013-08-25 15:13:28 UTC",
  "details": {},
  "dismissedDate": "null",
  "ignoredDate": null,
  "message": "A health check failed for the 'LDAP' connection, please see the MWS
health details page for more information.",
  "objectId": "ldap",
  "objectType": "Health",
  "observedDate": "2013-08-30 18:11:15 UTC",
  "origin": "MWS/HealthNotificationJob",
  "user": "admin",
  "id": "5230ed82e4b065347016d60d"
}
```

## Dismissing notifications

The HTTP PUT method is used to dismiss **Notifications**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/notifications/dismiss?api-version=3
PUT http://localhost:8080/mws/rest/notifications/<id>/dismiss?api-version=3
```

### Dismiss all notifications

#### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notifications/dismiss?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response

```
{"messages":["Updated 10 Notification objects"]}
```

### Dismiss single notification

#### URLs and parameters

```
PUT http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/dismiss?api-version=3
```

Parameter	Required	Type	Description
<b>id</b>	Yes	String	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
PUT http://localhost:8080/mws/rest/notifications/5230ed82e4b065347016d60d/dismiss?api-version=3
```

```
{
  "conditionId": "521a1f18e4b0e3f9031f47f5",
  "createdDate": "2013-08-25 15:13:28 UTC",
  "details": {},
  "dismissedDate": "2013-09-17 15:34:36 UTC",
  "ignoredDate": null,
  "message": "A health check failed for the 'LDAP' connection, please see the MWS health details page for more information.",
  "objectId": "ldap",
  "objectType": "Health",
  "observedDate": "2013-08-30 18:11:15 UTC",
  "origin": "MWS/HealthNotificationJob",
  "user": "admin",
  "id": "5230ed82e4b065347016d60d"
}
```

### Related topics

- [Resources introduction on page 1446](#)
- [Events on page 1528](#)
- [Fields: Events on page 1894](#)
- [Notifications on page 1585](#)
- [Fields: Notifications on page 2195](#)
- [Creating events and notifications on page 1704](#)
- [Plugin event service on page 1755](#)
- [Handling events on page 1711](#)

- [System events](#) on page 1444
- [Securing the connection with the message queue](#) on page 1417

## Permissions

This section describes behavior of the **Permissions** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

 The [Fields: User's Permissions](#) reference section contains the type and description of fields that all **Permissions** have in common.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/permissions</a>	<a href="#">Get all permissions</a>	--	<a href="#">Create single permission</a>	--
<a href="#">/rest/permissions/&lt;id&gt;</a>	<a href="#">Get single permission</a>	--	--	<a href="#">Delete single permission</a>
<a href="#">/rest/permissions/users/&lt;id&gt;</a>	<a href="#">Get a user's permissions</a>	--	--	--
<a href="#">/rest/permissions/users</a>	<a href="#">Get a current user's permissions</a>	--	--	--

This topic contains these sections:

- [Getting permissions](#) on page 1594
  - [Get all permissions](#) on page 1594
  - [Get single permission](#) on page 1595
  - [Get a user's permissions](#) on page 1595
  - [Get a current user's permissions](#) on page 1596
- [Creating permissions](#) on page 1597
  - [Create single permission](#) on page 1597
- [Deleting permissions](#) on page 1598
  - [Delete single permission](#) on page 1598

## Getting permissions

The HTTP GET method is used to retrieve **Permission** information. You can query all objects or a single object.

### Quick reference

```
GET http://localhost:8080/mws/rest/permissions?api-version=3
GET http://localhost:8080/mws/rest/permissions/<id>?api-version=3
```

## Get all permissions

### URLs and parameters

```
GET http://localhost:8080/mws/rest/permissions?api-version=3[&query={"field":"value"}
&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query permissions by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={ "type": "CUSTOM"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name": -1}

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/permissions?api-
version=3&fields=resource,action,description
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [{
    "resource" : "chart",
    "action" : "read",
    "description" : "The permission to view all charts."
  } ]
}
```

### Sorting and querying

See the sorting and querying sections of [Global URL parameters on page 1425](#).

## Get single permission

### URLs and parameters

```
GET http://localhost:8080/mws/rest/permissions/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the permission.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/permissions/<id>?api-version=3
```

```
{
  "action" : "create",
  "description" : "The permission to create all charts.",
  "id" : "50296335e4b0011b0f8394ec",
  "label" : "Create Chart",
  "resource" : "chart",
  "resourceFilter" : null,
  "type" : "custom",
  "scope" : "NONE",
  "version" : 0
}
```



For permissions with type "domain", scope must be GLOBAL or TENANT. All other permissions should have scope NONE.

## Get a user's permissions

### URLs and parameters

```
GET http://localhost:8080/mws/rest/permissions/users/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the user.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/permissions/users/bob?api-version=3
```

```
[
  {
    "action": "read",
    "description": "The permission to read all charts",
    "id": "5033b842e4b09cc61bedb818",
    "label": "",
    "resource": "chart",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  },
  {
    "action": "read",
    "description": "The permission to read all pages",
    "id": "5033b8a5e4b09cc61bedb82d",
    "label": "",
    "resource": "page",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  },
  {
    "action": "update",
    "description": "The permission to update all pages",
    "id": "5033b8a5e4b09cc61bedb82f",
    "label": "",
    "resource": "page",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  }
]
```

## Get a current user's permissions

### URLs and parameters

```
GET http://localhost/mws/rest/permissions/users/?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost/mws/rest/permissions/users/?api-version=3
```

```
[
  {
    "action": "read",
    "description": "The permission to read all charts",
    "id": "5033b842e4b09cc61bedb818",
    "label": "",
    "resource": "chart",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  },
  {
    "action": "read",
    "description": "The permission to read all pages",
    "id": "5033b8a5e4b09cc61bedb82d",
    "label": "",
    "resource": "page",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  },
  {
    "action": "update",
    "description": "The permission to update all pages",
    "id": "5033b8a5e4b09cc61bedb82f",
    "label": "",
    "resource": "page",
    "resourceFilter": null,
    "type": "custom",
    "scope": "NONE",
    "version": 1
  }
]
```

## Creating permissions

The HTTP POST method is used to create **Permissions**.

### Quick reference

```
POST http://localhost:8080/mws/rest/permissions?api-version=3
```

### Create single permission

#### URLs and parameters

```
POST http://localhost:8080/mws/rest/permissions?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

**i** The **resource**, **action**, and **type** are required on each permission.

Api permissions are permissions with the type 'api' and are the only permissions enforced by MWS.

Api permissions must map to a valid resource. For example, "services" is valid because there is a resource `/mws/rest/services`.

Api permissions must have `create`, `read`, `update`, or `delete` as the action.

The following is an example request body to create a permission:

```
POST http://localhost:8080/mws/rest/permissions?api-version=3
-----
{
    "resource" : "Chart",
    "action" : "read",
    "type" : "custom",
    "scope" : "NONE",
    "label" : "Read all charts",
    "description" : "The permissions to view all charts."
}
```

Sample response

If the request was successful, the response body is the new permission that was created exactly as shown in [Get single permission](#). On failure, the response is an error message.

## Deleting permissions

The HTTP DELETE method is used to delete **Permissions**.

Quick reference

```
DELETE http://localhost:8080/mws/rest/permissions/<id>?api-version=3
```

### Delete single permission

URLs and parameters

```
DELETE http://localhost:8080/mws/rest/permission/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the permission.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
JSON response
-----
{}

```

Related topics

- [Fields: User's Permissions on page 2428](#)
- [Resources introduction on page 1446](#)

## Plugins

This section describes behavior of the **Plugins** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Plugins](#) reference contains the type and description of all fields in the **Plugin** object. It also contains details regarding which fields are valid during PUT and POST actions.

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/plugins</b>	<a href="#">Get all plugins</a>	--	<a href="#">Create plugin</a>	--
<b>/rest/plugins/reporting-jobs/&lt;jobName&gt;?api-version=3</b>	<a href="#">Get all plugins reporting object</a>	--	--	--
<b>/rest/plugins/reporting-nodes/&lt;nodeName&gt;?api-version=3</b>	<a href="#">Get all plugins reporting object</a>	--	--	--
<b>/rest/plugins/reporting-vms/&lt;vmName&gt;?api-version=3</b>	<a href="#">Get all plugins reporting object</a>	--	--	--
<b>/rest/plugins/&lt;id&gt;</b>	<a href="#">Get single plugin</a>	<a href="#">Modify plugin</a>	--	<a href="#">Delete plugin</a>
<b>/rest/plugins/&lt;id&gt;/poll</b>	--	--	<a href="#">Trigger plugin poll</a>	--

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/plugins/&lt;id&gt;/services/&lt;serviceName&gt;</a>	<a href="#">Access a plugin web service</a>			

This topic contains these sections:

- [Getting plugins on page 1600](#)
  - [Get all plugins on page 1600](#)
  - [Get all plugins reporting object on page 1601](#)
  - [Get single plugin on page 1602](#)
- [Creating plugins on page 1602](#)
  - [Create plugin on page 1602](#)
- [Modifying plugins on page 1603](#)
  - [Modify plugin on page 1603](#)
  - [Trigger plugin poll on page 1604](#)
- [Deleting plugins on page 1605](#)
  - [Delete plugin on page 1605](#)
- [Accessing Plugin Web Services on page 1605](#)
  - [Access a plugin web service on page 1606](#)

## Getting plugins

The HTTP GET method is used to retrieve **Plugin** information. Queries for all objects, a single object, and query by reported object are available.

### Quick reference

```
GET http://localhost:8080/mws/rest/plugins?api-version=3
GET http://localhost:8080/mws/rest/plugins/<id>?api-version=3
GET http://localhost:8080/mws/rest/plugins/reporting-jobs/<jobName>?api-version=3
GET http://localhost:8080/mws/rest/plugins/reporting-nodes/<nodeName>?api-version=3
GET http://localhost:8080/mws/rest/plugins/reporting-vm/<vmName>?api-version=3
```

## Get all plugins

### URLs and parameters

```
GET http://localhost:8080/mws/rest/plugins?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/plugins?api-version=3&fields=id
-----
{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {"id": "plugin1"},
    {"id": "plugin2"},
    {"id": "plugin3"}
  ]
}
```

## Get all plugins reporting object

### URLs and parameters

```
GET http://localhost:8080/mws/rest/plugins/reporting-jobs/<jobName>?api-version=3
GET http://localhost:8080/mws/rest/plugins/reporting-nodes/<nodeName>?api-version=3
GET http://localhost:8080/mws/rest/plugins/reporting-vm/<vmName>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>jobName</b>	Yes	String	--	The name of the job to query by.
<b>nodeName</b>	Yes	String	--	The name of the node to query by.
<b>vmName</b>	Yes	String	--	The name of the VM to query by.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

This built-in query returns the same information as [Get all plugins](#), but filters the items to only plugins that are currently reporting the specified job, node, or VM (see [Reporting state data on page 1698](#)). The list is sorted ascending by the `precedence` field. In other words, the most authoritative plugin for the report is listed first. For more information, see [Data consolidation on page 1677](#).

```
GET http://localhost:8080/mws/rest/plugins/reporting-nodes/node1?api-
version=3&fields=id
-----
{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {"id": "plugin1"},
    {"id": "plugin2"},
    {"id": "plugin3"}
  ]
}
```

## Get single plugin

### URLs and parameters

```
GET http://localhost:8080/mws/rest/plugins/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response

```
{
  "id": "plugin1",
  "pluginType": "Native",
  "pollInterval": 30,
  "autoStart": true,
  "config": {
    "getJobs": "exec:///opt/moab/tools/workload.query.pl"
  },
  "state": "STARTED",
  "nextPollDate": "2011-12-02 17:28:52 UTC",
  "lastPollDate": "2011-12-02 17:28:22 UTC"
}
```

## Creating plugins

The HTTP POST method is used to create **Plugins**.

### Quick reference

```
POST http://localhost:8080/mws/rest/plugins?api-version=3
```

## Create plugin

### URLs and parameters

```
POST http://localhost:8080/mws/rest/plugins?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

When creating a plugin, the `id` and `pluginType` fields are required. The request body below shows all fields that are available when creating a plugin, along with some sample values.

JSON request body

```

{
  "id": "plugin1",
  "pluginType": "Native",
  "pollInterval": 30,
  "autoStart": true,
  "config": {
    "getJobs": "exec:///opt/moab/tools/workload.query.pl"
  }
}

```

### Sample response

JSON response for successful POST

```

{"id": "plugin1"}

```

### Restrictions

While it is possible to create a plugin with arbitrary nested configuration, such as:

```

...
"config": {
  "nestedObject": {
    "property1": "value1",
    "property2": "value2"
  },
  "nestedList": ["listItem1", "listItem2"]
}

```

It is *not* recommended, because the user interface (see [Plugin management on page 1742](#)) does not support editing or viewing any configuration data values other than strings.

## Modifying plugins

The HTTP PUT method is used to modify **Plugins**. Additionally, the POST method may be used to trigger an immediate poll of a **Plugin**.

### Quick reference

```

PUT http://localhost:8080/mws/rest/plugins/<id>?api-version=3
POST http://localhost:8080/mws/rest/plugins/<id>/poll?api-version=3

```

## Modify plugin

### URLs and parameters

```

PUT http://localhost:8080/mws/rest/plugins/<id>?api-version=3

```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

The request body below shows all the fields that are available when modifying a **Plugin**, along with some sample values.

```

JSON request body for plugin modification
-----
{
  "state":"STARTED",
  "pollInterval":30,
  "autoStart":true,
  "config":{
    "getJobs":"exec:///opt/moab/tools/workload.query.pl"
  },
  "state":"STARTED"
}
    
```

Sample response

```

JSON response
-----
{"messages":["Plugin plugin1 updated", "Started Plugin 'plugin1'"]}
    
```

**Trigger plugin poll**

URLs and parameters

```

POST http://localhost:8080/mws/rest/plugins/<id>/poll?api-version=3
    
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

Trigger poll

This resource call will trigger an immediate poll of the specified plugin. It is equivalent to the same operation on [Monitoring and lifecycle controls on page 1745](#).

Request body

No request body is required.

### Sample response

JSON response

```
-----  
{"messages":["Polled Plugin with ID 'myPlugin'"]}
```

## Deleting plugins

The HTTP DELETE method is used to delete **Plugins**.

### Quick reference

```
DELETE http://localhost:8080/mws/rest/plugins/<id>?api-version=3
```

### Delete plugin

#### URLs and parameters

```
DELETE http://localhost:8080/mws/rest/plugins/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response for successful DELETE

```
-----  
{}
```



Additional information about a successful DELETE can be found in the HTTP response header `X-MWS-Message`.

JSON response for an unsuccessful DELETE

```
-----  
{"messages":["Plugin plugin1 could not be deleted", "Error message describing the problem"]}
```

## Accessing Plugin Web Services

All HTTP methods can be used to access **Plugin Web Services**. However, some services only support specific methods. Check the specific plugin type documentation for more information.

Quick reference

```
GET http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
POST http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
PUT http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
DELETE http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
```

**Access a plugin web service**

URLs and parameters

```
GET http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
POST http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
PUT http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
DELETE http://localhost:8080/mws/rest/plugins/<id>/services/<serviceName>
[</objectId>]?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.
<b>objectId</b>	No	String	--	An arbitrary ID parameter that will be passed to the web service.
<b>serviceName</b>	Yes	String	--	The name of the web service, either in CamelCase or hyphenated.

See [Global URL parameters on page 1425](#) for available URL parameters.

Web service IDs

Translation is done to map [CamelCase](#) web service names to hyphenated names in the URL. For example, a web service method named `notifyEvent` on a plugin with a name of `notifications` may be called with the following URLs:

```
// CamelCase
/rest/plugins/notifications/services/notifyEvent

// Hyphenated
/rest/plugins/notifications/services/notify-event
```

HTTP method and request body

Because plugin custom web services do not need to distinguish which HTTP method is used (see [Custom web services on page 1676](#)), it is recommended to use GET and POST when making requests to access

web services unless documented otherwise. The request body and output may vary for each web service called. See [Plugin types on page 1607](#) for the requested plugin for available web services, request parameters, and expected output.

#### Related topics

- [Fields: Plugins on page 2197](#)
- [Resources introduction on page 1446](#)
- [Plugin types on page 1607](#)

## Plugin types

This section describes behavior of the **Plugin Type** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Plugin Types](#) reference section contains the type and description of all fields in the **Plugin Type** object. It also contains details regarding which fields are valid during PUT and POST actions.

#### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/plugin-types</a>	<a href="#">Get all plugin types</a>	<a href="#">Creating or updating plugin types</a>	--	--
<a href="#">/rest/plugin-types/&lt;id&gt;</a>	<a href="#">Get single plugin type</a>	--	--	--

This topic contains these sections:

- [Getting plugin types on page 1607](#)
  - [Get all plugin types on page 1608](#)
  - [Get single plugin type on page 1608](#)
- [Creating or updating plugin types on page 1609](#)
  - [Update plugin type \(file\) on page 1609](#)
  - [Update plugin type \(JAR\) on page 1611](#)

### Getting plugin types

The HTTP GET method is used to retrieve **Plugin Type** information. Queries for all objects and a single object are available.

### [Quick reference](#)

```
GET http://localhost:8080/mws/rest/plugin-types/<id>?api-version=3
```

## Get all plugin types

### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/plugin-types?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### [Sample response](#)

```
GET http://localhost:8080/mws/rest/plugin-types?api-version=3&fields=id
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {"id": "vCenter"},
    {"id": "Native"}
  ]
}
```

## Get single plugin type

### [URLs and parameters](#)

```
GET http://localhost:8080/mws/rest/plugin-types/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response

```

{
  "author": "Adaptive Computing Enterprises, Inc.",
  "commonsVersion": "0.9.3 > *",
  "description": "Polls a VMware® vCenter™ Server for information on the hypervisors
and virtual machines it manages.",
  "documentationLink": "",
  "email": "",
  "eventComponent": 1,
  "realizedEventComponent": 513,
  "id": "VCenter",
  "initialPlugins": { },
  "instances": [
    {"id":"vcenter"}
  ],
  "issueManagementLink": "",
  "license": "APACHE",
  "mwsVersion": "7.1.2 > *",
  "pollMethod": true,
  "scmLink": "",
  "title": "VCenter",
  "version": "1.0",
  "webServices": [ ],
  "website": "http://www.adaptivecomputing.com"
}

```

## Creating or updating plugin types

The HTTP PUT method is used to create or update **Plugin Types**. The Content-Type HTTP header is used to determine if the request contains a single class file as plaintext or the binary data of a JAR file. Each request is explained in the following sections.

### Quick reference

```
PUT http://localhost:8080/mws/rest/plugin-types?api-version=3[&reload-plugins=false]
```



There is a known issue with dynamically updating plugin types with typed field injection. For more information, see [Add or update plugin types on page 1739](#).

## Update plugin type (file)

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/plugin-types?api-version=3[&reload-plugins=false]
```

Parameter	Required	Type	Valid values	Description
<b>reload-plugins</b>	No	String	true or false	Reloads all plugins of this type on successful update. Defaults to true.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

This function is idempotent, meaning it will create the **Plugin Type** if it does not exist or update it if it does. The request body is the actual contents of the class file to upload. This web service is an exception to most as it *requires* a content type of `application/x-groovy` or `text/plain`.



If the `application/x-groovy` or `text/plain` content types are not used in the request, it will be interpreted as JSON, resulting in a failure.

Plaintext upload

```

package test

import com.adaptc.mws.plugins.*

class UploadPlugin {
    static author = "Adaptive Computing"
    static description = "A sample plugin class"
    String id

    public void configure() throws InvalidPluginConfigurationException {
        def myConfig = config
        def errors = []
        if (!myConfig.arbitraryKey)
            errors << "Missing arbitraryKey!"
        if (errors)
            throw new InvalidPluginConfigurationException(errors)
    }

    public def customService(Map params) {
        return params
    }
}

```



If using the [curl](#) library to perform plugin type uploading, the equivalent of the command-line option `--data-binary` must be used to send the request body. Otherwise compilation errors may be encountered when uploading the plugin type.

### Sample response

The response of this task is the same as the [Get all plugin types](#) task. The reason that the return of this task is a list is to accommodate the possibility of uploading multiple plugin types in a single JAR file as explained in the next section.

## Update plugin type (JAR)

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/plugin-types?api-version=3&jar-
filename=<filename.jar>[&reload-plugins=false]
```

Parameter	Required	Type	Valid values	Description
<b>jar-file-name</b>	Yes	String	--	The filename of the JAR file that is being uploaded.
<b>reload-plugins</b>	No	String	true or false	Reloads all plugins of this type on successful update. Defaults to true.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

This function is idempotent, meaning it will create the **Plugin Types** if they do not exist or update them if they do. The request body is the binary contents of the JAR file to upload. This web service is an exception to most as it *requires* a content type of `application/x-jar`.



If the `application/x-jar` content type is not used in the request, it will be interpreted as JSON, resulting in a failure.



If using the [curl](#) library to perform plugin type uploading, the equivalent of the command-line option `--data-binary` must be used to send the request body. Otherwise compilation errors may be encountered when uploading the plugin type.

### Sample response

The response of this task is the same as the [Get all plugin types](#) task. Note that when using a JAR file, multiple plugin types may be uploaded in the same request.

### Related topics

- [Fields: Plugin Types on page 2204](#)
- [Resources introduction on page 1446](#)
- [Plugins on page 1599](#)

## Policies

This section describes behavior of the **Policies** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The **Fields: Policies** reference section contains the type and description of fields of all **Policies**.

### Supported policies

Name	ID
Automatic VM Migration	auto-vm-migration
Fairshare	fairshare
Hypervisor Allocation Overcommit	hv-allocation-overcommit
Migration Exclusion List	migration-exclusion-list
Node Allocation	node-allocation

### Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/policies</b>	<a href="#">Get all policies</a>	--	--	--
<b>/rest/policies/&lt;id&gt;</b>	<a href="#">Get single policy</a>	<a href="#">Modify policy</a>	--	--

This topic contains these sections:

- [Getting policies on page 1612](#)
  - [Get all policies on page 1613](#)
  - [Get single policy on page 1613](#)
- [Modifying policies on page 1615](#)
  - [Modify policy on page 1616](#)

## Getting policies

The HTTP GET method is used to retrieve **Policies** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/policies?api-version=3
```

## Get all policies

### URLs and parameters

```
GET http://localhost:8080/mws/rest/policies?api-version=3
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Query for specific results.	query={"state":"DISABLED","-conflicted":"false"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"id":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/policies?api-version=3&fields=id,state,conflicted
```

```
{
  "totalCount": 4,
  "resultCount": 4,
  "results": [ {
    "conflicted": false,
    "state": "DISABLED",
    "id": "auto-vm-migration"
  }, {
    "conflicted": false,
    "state": "DISABLED",
    "id": "hv-allocation-overcommit"
  }, {
    "conflicted": false,
    "state": "DISABLED",
    "id": "node-allocation"
  }, {
    "conflicted": false,
    "state": "DISABLED",
    "id": "migration-exclusion-list"
  }
]
```

## Get single policy

### URLs and parameters

```
GET http://localhost:8080/mws/rest/policies/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample responses

#### Auto VM Migration

```
{
  "conflicted": false,
  "description": "Controls how virtual machines are automatically migrated.",
  "id": "auto-vm-migration",
  "name": "Auto VM Migration",
  "potentialConflicts": [],
  "priority": 1,
  "state": "DISABLED",
  "tags": [],
  "types": [],
  "version": 0,
  "genericMetricThresholds": {
    "GMETRIC1": 1.3
  },
  "processorUtilizationThreshold": 0.5,
  "memoryUtilizationThreshold": 0.4
}
```

#### Fairshare

```
{
  "conflicted": false,
  "decayFactor": 0.44,
  "depth": 4,
  "description": "Control job feasibility and priority decisions based on system utilization targets for users, groups, accounts, classes, and QoS levels.",
  "intervalSeconds": 600,
  "name": "Fairshare",
  "potentialConflicts": [],
  "priority": 16,
  "state": "ENABLED",
  "tags": [],
  "types": [],
  "usageMetric": "DEDICATED_PROCESSOR_SECONDS_DELIVERED",
  "version": 3,
  "id": "fairshare"
}
```

## Hypervisor Allocation Overcommit

```

{
  "conflicted": false,
  "description": "Controls how hypervisors are overallocated with regards to
processors and memory.",
  "id": "hv-allocation-overcommit",
  "name": "Hypervisor Allocation Overcommit",
  "potentialConflicts": [],
  "priority": 2,
  "state": "DISABLED",
  "tags": [],
  "types": [],
  "version": 0,
  "processorAllocationLimit":29.5,
  "memoryAllocationLimit":1.2
}

```

## Migration Exclusion List

```

{
  "conflicted": false,
  "description": "Controls which machines are excluded from automatic live migration
operations.",
  "hvExclusionList": ["blade05", "blade02"],
  "name": "Migration Exclusion List",
  "potentialConflicts": [],
  "priority": 100,
  "state": "DISABLED",
  "tags": [],
  "types": [],
  "version": 1,
  "vmExclusionList": ["vm1", "vm5"],
  "id": "migration-exclusion-list"
}

```

## Node Allocation

```

{
  "conflicted": false,
  "description": "Controls how nodes are selected for workload placement.",
  "id": "node-allocation",
  "name": "Node Allocation",
  "potentialConflicts": [],
  "priority": 3,
  "state": "DISABLED",
  "tags": [],
  "types": [],
  "version": 0,
  "nodeAllocationAlgorithm": "CustomPriority",
  "customPriorityFunction": "-100*GMETRIC[vmcount]"
}

```

## Modifying policies

The HTTP PUT method is used to modify **Policies**.

[Quick reference](#)

```
PUT http://localhost:8080/mws/rest/policies/<id>?api-version=3
```

**Modify policy**

[URLs and parameters](#)

```
PUT http://localhost:8080/mws/rest/policies/<id>?api-version=3 [&change-mode=set]
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

[Additional URL parameters](#)

URL parameters for modifying a Migration Exclusion Lists Policy.

Migration Exclusion Lists parameter	Required	Type	Valid values	Description
<b>change-mode</b>	No	String	set (default) add remove	If <b>set</b> , replace existing exclusion list(s) with the given one. If <b>add</b> , add the given VMs/HVs to the existing exclusion list(s). If <b>remove</b> , remove the given VMs/HVs from the existing exclusion list(s).

[Request body](#)

In general, the fields shown in the [Fields: Policies](#) reference section are *not* available for modification. However, the `state` field may be modified to a valid `PolicyState`. All other fields listed in the specific policy type sections (i.e. `AutoVMMigrationPolicy`) may be modified unless documented otherwise.

- The request body below shows all the fields that are available when modifying a Auto VM Migration Policy, along with some sample values.

```
JSON request body for Auto VM Migration Policy
{
  "genericMetricThresholds": {
    "GENERICTHRESHOLD": 5
  },
  "memoryUtilizationThreshold": 0.5,
  "processorUtilizationThreshold": 0.4
}
```

- The request body below shows all the fields that are available when modifying a Fairshare Policy, along with some sample values.

```
JSON request body for Fairshare Policy
-----
{
  "decayFactor": 0.44,
  "depth": 4,
  "intervalSeconds": 600,
  "usageMetric": "DEDICATED_PROCESSOR_SECONDS_DELIVERED",
}
```

- The request body below shows all the fields that are available when modifying a Hypervisor Allocation Overcommit Policy, along with some sample values.

```
JSON request body for Hypervisor Allocation Overcommit Policy
-----
{
  "processorAllocationLimit":29.5,
  "memoryAllocationLimit":1.2
}
```

- The request body below shows all the fields that are available when modifying a Migration Exclusion Lists Policy, along with some sample values.

```
JSON request body for Migration Exclusion Lists Policy
-----
{
  "vmExclusionList" : ["vm1","vm3","vm5"],
  "hvExclusionList" : ["hv2","hv3","hv6"]
}
```

- The request body below shows all the fields that are available when modifying a Node Allocation Policy, along with some sample values.

```
JSON request body for Node Allocation Policy
-----
{
  "nodeAllocationAlgorithm" : "CustomPriority",
  "customPriorityFunction" : "-100*GMETRIC[vmcount]"
}
```

### Sample response

```
JSON response
-----
{
  "messages": ["Policy auto-vm-migration updated"]
}
```

### Samples

Enable the Auto VM Migration Policy and set values.

```
PUT http://localhost:8080/mws/rest/policies/auto-vm-migration?api-version=3
```

```
{
  "state": "enabled",
  "migrationAlgorithmType": "overcommit",
  "processorUtilizationThreshold": 0.5,
  "memoryUtilizationThreshold": 0.4
}
```

 As noted in the [Fields: Policies](#) reference section documentation for `AutoVMMigrationPolicy`, if the `state` is set to `ENABLED`, then the `migrationAlgorithmType` must *not* be set to `NONE`.

### Restrictions

All policies:

- Fields cannot be modified while the policy is disabled. Enable the policy to modify the field.

Auto VM Migration

- Arbitrary metrics can be added to **genericMetricThresholds**, but they cannot be removed once added.
- The **migrationAlgorithmType** field cannot be modified while the policy is disabled. Enable the policy to modify the field.
- Moab is configured with a default limit of 10 generic metrics. If this limit is reached, such as when arbitrary metrics are added to **genericMetricThresholds**, the metric will not be reported. To increase this limit, set the `MAXGMETRIC` property in the Moab configuration file.

Fairshare

- Updating the **usageMetric** field will clear all credential-based fairshare interval data.

Related topics

- [Fields: Policies on page 2210](#)
- [Fairshare on page 1618](#)
- [Resources introduction on page 1446](#)

## Fairshare

This section describes behavior of the **Fairshare** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

 The supported methods table below requires each resource to be accessed with a URL parameter of `api-version=3`.

For more information, see [Requesting specific API versions on page 1428](#).

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/policies/fairshare</b>	<a href="#">Get all fairshare interval data on page 1620</a>	--	--	--
<b>/rest/policies/fairshare/&lt;credentialType&gt;</b>	<a href="#">Get all fairshare interval data for a single credential type on page 1623</a>	--	--	--
<b>/rest/policies/fairshare/&lt;credentialType&gt;/&lt;name&gt;</b>	<a href="#">Get all fairshare interval data for a single credential on page 1626</a>	--	--	--

This topic contains these sections:

- [Getting credential-based fairshare interval data on page 1619](#)
  - [Get all fairshare interval data on page 1620](#)
  - [Get all fairshare interval data for a single credential type on page 1623](#)
  - [Get all fairshare interval data for a single credential on page 1626](#)

## Getting credential-based fairshare interval data

The HTTP GET method is used to retrieve **Policies** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/policies/fairshare/credentials?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/accounts?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/classes?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/groups?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/qoses?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/users?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/accounts/<name>?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/classes/<name>?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/groups/<name>?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/qoses/<name>?api-version=3
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/users/<name>?api-version=3
```

## Get all fairshare interval data

### URLs and parameters

```
GET http://localhost:8080/mws/rest/policies/fairshare/credentials?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/policies/fairshare/credentials?api-version=3
```

```
{
  "totalCount": 4,
  "resultCount": 4,
  "results": [
    {
      "name": "jbethune",
      "target_type": null,
      "target": null,
      "interval_data": [
        0,
        0,
        0,
        0
      ],
      "credential_type": "USER"
    },
    {
      "name": "jfoote",
      "target_type": null,
      "target": null,
      "interval_data": [
        2104.16,
        2377.06,
        2240.1,
        2550
      ],
      "credential_type": "GROUP"
    },
    {
      "name": "NOGROUP",
      "target_type": null,
      "target": null,
      "interval_data": [
        0,
        0,
        0,
        0
      ],
      "credential_type": "GROUP"
    },
    {
      "name": "DEFAULT",
      "target_type": null,
      "target": null,
      "interval_data": [
        0,
        0,
        0,
        0
      ],
      "credential_type": "ACCOUNT"
    },
    {
      "name": "Administration",
      "target_type": null,
      "target": null,
      "interval_data": [
        5256.28,
        6247.05,
        6048.27,
        6948.67
      ],
      "credential_type": "ACCOUNT"
    }
  ]
}
```

```
]
}
```

## Get all fairshare interval data for a single credential type

### URLs and parameters

```
GET
http://localhost:8080/mws/rest/policies/fairshare/credentials/<accounts|classes|groups
|qoses|users>?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample responses

```
GET http://localhost:8080/mws/rest/policies/fairshare/credentials/accounts?api-version=3
```

```
-----
{
  "totalCount": 6,
  "resultCount": 6,
  "results": [
    {
      "name": "jbethune",
      "target_type": null,
      "target": null,
      "interval_data": [
        0,
        0,
        0,
        0
      ],
      "credential_type": "ACCOUNT"
    },
    {
      "name": "Administration",
      "target_type": null,
      "target": null,
      "interval_data": [
        5256.28,
        6247.05,
        6048.27,
        6948.67
      ],
      "credential_type": "ACCOUNT"
    },
    {
      "name": "Shared",
      "target_type": null,
      "target": null,
      "interval_data": [
        4261.38,
        4951.09,
        4480.2,
        5000.54
      ],
      "credential_type": "ACCOUNT"
    },
    {
      "name": "Engineering",
      "target_type": null,
      "target": null,
      "interval_data": [
        15034.64,
        17245.93,
        15008.67,
        17085
      ],
      "credential_type": "ACCOUNT"
    },
    {
      "name": "Test",
      "target_type": null,
      "target": null,
      "interval_data": [
        1808.08,
        1873.96,
        1568.07,
        1757.33
      ],
      "credential_type": "ACCOUNT"
    }
  ],
}
```

```

    {
      "name": "Research",
      "target_type": null,
      "target": null,
      "interval_data": [
        47606.8,
        52861.83,
        46370.07,
        52785
      ],
      "credential_type": "ACCOUNT"
    }
  ]
}

```

### Get all fairshare interval data for a single credential

#### URLs and parameters

```

GET
http://localhost:8080/mws/rest/policies/fairshare/credentials/<accounts|classes|groups|qoses|users>/<name>?api-version=3

```

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The unique name of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```

GET
http://localhost:8080/mws/rest/policies/fairshare/credentials/accounts/DEFAULT?api-version=3
-----
{
  "name": "DEFAULT",
  "target_type": null,
  "target": null,
  "interval_data": [
    0,
    0,
    0,
    0
  ],
  "credential_type": "ACCOUNT"
}

```

#### Related topics

- [Policies on page 1611](#)
- [Resources introduction on page 1446](#)

## Principals

This section describes behavior of the **Principal** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Principals](#) reference contains the type and description of all fields in the **Principal** object. It also contains details regarding which fields are valid during PUT and POST actions.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/principals</a>	<a href="#">Get all principals</a>	--	<a href="#">Create single principal</a>	--
<a href="#">/rest/principals/&lt;id&gt;</a>	<a href="#">Get single principal</a>	<a href="#">Modify single principal</a>	--	<a href="#">Delete single principal</a>
<a href="#">/rest/principals/&lt;name&gt;</a>	<a href="#">Get single principal</a>	<a href="#">Modify single principal</a>	--	<a href="#">Delete single principal</a>

This topic contains these sections:

- [Getting principals on page 1627](#)
  - [Get all principals on page 1628](#)
  - [Get single principal on page 1629](#)
- [Creating principals on page 1630](#)
  - [Create single principal on page 1630](#)
- [Modifying principals on page 1631](#)
  - [Modify single principal on page 1631](#)
- [Deleting principals on page 1633](#)
  - [Delete single principal on page 1633](#)

## Getting principals

The HTTP GET method is used to retrieve **Principal** information. You can query all objects or a single object.

Quick reference

```
GET http://localhost:8080/mws/rest/principals?api-version=3[&query={"field":"value"}
&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/principals/<id>?api-version=3
GET http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

**Get all principals**

URLs and parameters

```
GET http://localhost:8080/mws/rest/principals?api-version=3[&query={"field":"value"}
&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query principals by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={ "name":"Acme Principal"}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":- 1}

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/principals?api-version=3&fields=name,group
-----
{
  "totalCount": 2,
  "resultCount": 2,
  "results": [
    {
      "groups": [
        {
          "name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev",
          "type": "LDAPGROUP"
        }
      ],
      "name": "Engineering-Principal"
    },
    {
      "groups": [
        {
          "name": "CN=Marketing,CN=Users,DC=corp,DC=cloud,DC=dev",
          "type": "LDAPGROUP"
        }
      ],
      "name": "Marketing-Principal"
    }
  ]
}
```

Sorting and Querying

See the sorting and querying sections of [Global URL parameters on page 1425](#).

## Get single principal

### URLs and parameters

```
GET http://localhost:8080/mws/rest/principals/<id>?api-version=3
GET http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the principal.
<b>name</b>	Yes	String	--	The name of the principal.

 You must specify either id or name, but you do not have to specify both.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/principals/principal8?api-version=3

{
  "attachedRoles": [ {
    "description": "This is a role for normal users in the Acme BU Group.",
    "id": "5033b8eae4b09cc61bedb895",
    "name": "Acme-User-Role",
    "permissions": [
      {
        "action": "read",
        "description": "The permission to read all nodes",
        "id": "5033b842e4b09cc61bedb818",
        "label": "",
        "resource": "nodes",
        "resourceFilter": null,
        "type": "api",
        "version": 1
      }
    ],
    "version": 2
  } ],
  "description": "Principal 8",
  "groups": [ {
    "name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev",
    "type": "LDAPGROUP"
  } ],
  "id": "5033d33fe4b018b28745fecb",
  "name": "principal8",
  "users": [
    {
      "name": "jhammon",
      "type": "LDAP"
    },
    {
      "name": "bjones",
      "type": "LDAP"
    }
  ],
  "version": 0
}
```

## Creating principals

The HTTP POST method is used to submit **Principals**.

### Quick reference

```
POST http://localhost:8080/mws/rest/principals?api-version=3
```

### Create single principal

#### URLs and parameters

```
POST http://localhost:8080/mws/rest/principals?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

**i** The **name** field is required and must contain only letters, digits, periods, dashes, and underscores. The **attachedRoles** field expects an array of Role IDs *or* names:

The following is an example request body to create a principal:

```
POST http://localhost:8080/mws/rest/principals?api-version=3
-----
{
  "name" : "Acme-Principal",
  "attachedRoles" : [{"name":"Acme-User-Role"}],
  "description" : "A cool principal",
  "groups" : [{"name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev",
"type":"LDAPGROUP"}],
  "users" : [{
    "name" : "john",
    "type" : "LDAP"
  } ]
}
```

Sample response

If the request was successful, the response body is the new principal that was created, exactly as shown in [Get single principal](#). On failure, the response is an error message.

## Modifying principals

The HTTP PUT method is used to modify **Principals**.

Quick reference

```
PUT http://localhost:8080/mws/rest/principals/<id>?api-version=3
PUT http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

### Modify single principal

URLs and parameters

```
PUT http://localhost:8080/mws/rest/principals/<id>?api-version=3
PUT http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Principal.

Parameter	Required	Type	Valid values	Description
<b>name</b>	Yes	String	--	The name of the Principal. <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; margin-top: 5px;">  The <b>name</b> field must contain only letters, digits, periods, dashes, and underscores.                 </div>
<b>change-mode</b>	Yes	String	add remove set (default)	If <b>add</b> , add the given objects (ldapGroups, ldapOUs, etc.) to the objects that already exist. If <b>remove</b> , delete the given objects from the objects that already exist. If <b>set</b> , add the given objects (ldapGroups, ldapOUs, etc.) and remove the objects that already exist.

See [Global URL parameters on page 1425](#) for available URL parameters.

 You must specify either **id** or **name**, but you do not have to specify both.  
 The **attachedRoles** field expects an array of Role IDs *or* names:

Example request

```
PUT http://localhost/mws/rest/principals/Acme-Principal?api-version=3
{
  "groups" : [ {
    "name" : "CN=Marketing,CN=Users,DC=mycompany,DC=com",
    "type" : "LDAPGROUP"
  }, {
    "name" : "CN=Sales,CN=Users,DC=mycompany,DC=com",
    "type" : "LDAPGROUP"
  } ],
  "users" : [ {
    "name" : "jhammon",
    "type" : "LDAP"
  } ]
}
```

 The **version** field contains the current version of the database entry. This field cannot be updated directly. However, if **version** is included in the modify request, it will be used to verify that another client did not update the object between the time that the data was retrieved and the modify request was delivered.

Sample response

If the request was successful, the response body is the modified principal as shown in [Get single principal](#). On failure, the response is an error message.

## Deleting principals

The HTTP DELETE method is used to delete **Principals**.

### Quick reference

```
DELETE http://localhost:8080/mws/rest/principals/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

## Delete single principal

### URLs and parameters

```
DELETE http://localhost:8080/mws/rest/principals/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/principals/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the principal.
<b>name</b>	Yes	String	--	The name of the principal.

See [Global URL parameters on page 1425](#) for available URL parameters.



You must specify either **id** or **name**, but you do not have to specify both.

### Sample response

JSON response

```
{}
```

### Related topics

- [Fields: Principals on page 2249](#)
- [Resources introduction on page 1446](#)

## Priority

This section describes behavior of the **priority** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/priority</b>	<a href="#">Get all priorities on page 1634</a>	<a href="#">Modify priorities on page 1636</a>	--	--

This topic contains these sections:

- [Getting priorities on page 1634](#)
  - [Get all priorities on page 1634](#)
- [Modifying priorities on page 1635](#)
  - [Modify priorities on page 1636](#)

## Getting priorities

The HTTP GET method is used to retrieve **priority** information.

### Quick reference

```
GET http://localhost:8080/mws/rest/priority?api-version=3
```

### Get all priorities

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/priority?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```

{
  "service": {
    "weight": 1,
    "queue_time": 1,
    "x_factor": 0,
    "policy_violation": 0,
    "bypass": 0
  },
  "target": {
    "weight": 1,
    "queue_time": 0,
    "x_factor": 0
  },
  "credential": {
    "weight": 1,
    "user_credential": 0,
    "group_credential": 0,
    "account_credential": 0,
    "class_credential": 0,
    "qos_credential": 0
  },
  "attribute": {
    "weight": 1,
    "attribute": 0,
    "state": 0
  },
  "fairshare": {
    "weight": 1,
    "user_credential": 1000,
    "group_credential": 0,
    "account_credential": 0,
    "class_credential": 0,
    "qos_credential": 0,
    "jobs_per_user": 0,
    "processor_seconds_per_user": 0,
    "processors_per_user": 0
  },
  "resource": {
    "weight": 1,
    "node": 0,
    "disk": 0,
    "memory": 0,
    "swap": 0,
    "processor_equivalent_seconds": 0,
    "walltime": 0
  },
  "usage": {
    "weight": 1,
    "consumed": 0,
    "remaining": 0,
    "percentage_consumed": 0
  }
}

```

## Modifying priorities

The HTTP PUT method is used to update **priority** information.

Quick reference

```
PUT http://localhost:8080/mws/rest/priority?api-version=3
```

## Modify priorities

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/priority?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample body

```
PUT http://localhost:8080/mws/rest/priority?api-version=3
{
  "service": {
    "weight": 2,
    "queue_time": 2,
    "x_factor": 1,
    "policy_violation": 1,
    "bypass": 1
  }
}
```

### Related topics

- [Resources introduction on page 1446](#)

# Reports

This section describes behavior of the reporting framework in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Reports](#), [Fields: Report Samples](#), and [Fields: Report Datapoints](#) reference sections contain the type and description of all fields in the **Report**, **Sample**, and **Datapoint** objects. They also contains details regarding which fields are valid during PUT and POST actions.

### Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/reports</b>	<a href="#">Get all reports (no data)</a>	--	<a href="#">Create report</a>	<a href="#">Delete report</a>
<b>/rest/reports/&lt;name&gt;</b>	<a href="#">Get single report (with data)</a>	--	--	--
<b>/rest/reports/&lt;id&gt;</b>	<a href="#">Get single report (with data)</a>	--	--	--

Resource	GET	PUT	POST	DELETE
/rest/reports/<name>/datapoints	<a href="#">Get datapoints for single report</a>	--	--	--
/rest/reports/<id>/datapoints	<a href="#">Get datapoints for single report</a>	--	--	--
/rest/reports/<name>/samples	<a href="#">Get samples for report</a>	--	<a href="#">Create samples for report</a>	--
/rest/reports/<id>/samples	<a href="#">Get samples for report</a>	--	<a href="#">Create samples for report</a>	--

This topic contains these sections:

- [Getting reports on page 1637](#)
  - [Get all reports \(no data\) on page 1638](#)
  - [Get single report \(with data\) on page 1639](#)
  - [Get datapoints for single report on page 1640](#)
- [Getting samples for reports on page 1641](#)
  - [Get samples for report on page 1642](#)
- [Creating reports on page 1642](#)
  - [Create report on page 1643](#)
- [Creating samples on page 1644](#)
  - [Create samples for report on page 1644](#)
- [Deleting reports on page 1645](#)
  - [Delete report on page 1645](#)

## Getting reports

The HTTP GET method is used to retrieve **Report** information. Queries for all reports with no attached data and a single report with associated data are available.

### Quick reference

```
GET http://localhost:8080/mws/rest/reports?api-version=3[&query={"field":"value"}
&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/reports/<id>?api-version=3
GET http://localhost:8080/mws/rest/reports/<name>?api-version=3
```

## Get all reports (no data)

### URLs and parameters

```
GET http://localhost:8080/mws/rest/reports?api-version=3[&query={"field":"value"}
&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Description	Example
<b>query</b>	No	JSON	Queries for specific results. It is possible to query reports by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"reportSize":4}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response

```
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "id": "3efe5c670be86ba8560397ff",
    "name": "cpu-util"
    ...
  } ]
}
```

Samples

```
GET http://localhost:8080/mws/rest/reports?api-version=3&fields=id,name

{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {
      "id": "3efe5c670be86ba8560397ff",
      "name": "cpu-util"
    },
    {
      "id": "3efe5c670be86ba856039800",
      "name": "cpu-temp"
    },
    {
      "id": "3efe5c670be86ba856039801",
      "name": "cpu-load"
    }
  ]
}
```

**Get single report (with data)**

URLs and parameters

```
GET http://localhost:8080/mws/rest/reports/<id>?api-version=3
GET http://localhost:8080/mws/rest/reports/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the report.
<b>name</b>	Yes	String	--	The name of the report.

**i** Only one of **id** or **name** are required.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

In the example below, the first datapoint has a null data element, which means that the `minimumSampleSize` configured for the report was not met when consolidating the datapoint. The second datapoint contains actual data.

```

JSON response
-----
{
  "consolidationFunction": "average",
  "datapointDuration": 15,
  "datapoints": [
    {
      "endDate": "2011-12-02 17:28:22 UTC",
      "startDate": "2011-12-02 17:28:22 UTC",
      "firstSampleDate": null,
      "lastSampleDate": null,
      "data": null
    },
    {
      "endDate": "2011-12-02 17:28:23 UTC",
      "startDate": "2011-12-02 17:28:37 UTC",
      "firstSampleDate": "2011-12-02 17:28:23 UTC",
      "lastSampleDate": "2011-12-02 17:28:30 UTC",
      "data": {
        "utilization": 99.89,
        "time": 27.433333333333337
      }
    }
  ],
  "description": "Example of CPU utilization reporting",
  "id": "3efe5c670be86ba8560397ff",
  "keepSamples": false,
  "minimumSampleSize": 1,
  "name": "cpu-util",
  "reportSize": 2
}

```

### Get datapoints for single report

URLs and parameters

```

GET http://localhost:8080/mws/rest/reports/<id>/datapoints?api-version=3 [&query=
{"field": "value"}&sort={"field": <1|-1>}]
GET http://localhost:8080/mws/rest/reports/<name>/datapoints?api-version=3 [&query=
{"field": "value"}&sort={"field": <1|-1>}]

```

Parameter	Required	Type	Description	Example
<b>id</b>	Yes	String	The unique identifier of the report.	--
<b>name</b>	Yes	String	The name of the report.	--
<b>query</b>	No	JSON	Queries for specific results.	query={"reportSize":4}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

**i** Only one of **id** or **name** are required.

It is possible to query reports by one or more fields based on [MongoDB query syntax](#).

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

This function is exactly the same as [Get single report \(with data\)](#). No report metadata (i.e. description, minimumSampleSize, etc.) is returned.

JSON response

```

{
  "resultCount":1,
  "totalCount":1,
  "results":[
    {
      "endDate": "2011-12-02 17:28:22 UTC",
      "startDate": "2011-12-02 17:28:22 UTC",
      "firstSampleDate": null,
      "lastSampleDate": null,
      "data": null
    },
    {
      "endDate": "2011-12-02 17:28:37 UTC",
      "startDate": "2011-12-02 17:28:37 UTC",
      "firstSampleDate": "2011-12-02 17:28:23 UTC",
      "lastSampleDate": "2011-12-02 17:28:23 UTC",
      "data": {
        "utilization": 99.89,
        "time": 27.433333333333337
      }
    }
  ]
}

```

## Getting samples for reports

The HTTP GET method is used to retrieve **Sample** information.

### Quick reference

```

GET http://localhost:8080/mws/rest/reports/<id>/samples?api-version=3[&query=
{"field":"value"}&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/reports/<name>/samples?api-version=3[&query=
{"field":"value"}&sort={"field":<1|-1>}]

```

## Get samples for report

### URLs and parameters

```
GET http://localhost:8080/mws/rest/reports/<id>/samples?api-version=3[&query=
{"field":"value"}&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/reports/<name>/samples?api-version=3[&query=
{"field":"value"}&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Description	Example
<b>id</b>	Yes	String	The unique identifier of the report.	--
<b>name</b>	Yes	String	The name of the report.	--
<b>query</b>	No	JSON	Queries for specific results.	query={"reports-ize":4}
<b>sort</b>	No	JSON	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

**i** Only one of **id** or **name** are required.

It is possible to query reports by one or more fields based on [MongoDB query syntax](#).

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
JSON response
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "timestamp": "2011-12-02 17:28:37 UTC"
    "data":{
      "cpu1":2.3,
      "cpu2":1.2,
      "cpu3":0.0,
      "cpu4":12.1
    }
  },
  ...
  ]
}
```

## Creating reports

The HTTP POST method is used to create **Reports**. Operations are available to create reports with or without historical datapoints.

### Quick reference

```
POST http://localhost:8080/mws/rest/reports?api-version=3
```

## Create report

### URLs and parameters

```
POST http://localhost:8080/mws/rest/reports?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

To create a report, several fields are required as documented in [Fields: Reports on page 2261](#).

The request body below shows all the fields that are available during report creation.

JSON request body

```
{
  "name": "cpu-util",
  "description": "An example report on cpu utilization",
  "consolidationFunction": "average",
  "datapointDuration": 15,
  "minimumSampleSize": 1,
  "reportSize": 2,
  "keepSamples": true,
  "reportDocumentSize": 1024,
  "datapoints": [
    {
      "startDate": "2011-12-01 19:16:57 UTC",
      "endDate": "2011-12-01 19:16:57 UTC",
      "data": {
        "time": 30,
        "util": 99.98
      }
    }
  ]
}
```

### Sample response

```
{
  "messages": ["Report cpu-util created"],
  "id": "3efe5c670be86ba8560397ff",
  "name": "cpu-util"
}
```

Samples

```
POST http://localhost:8080/mws/rest/reports?api-version=3 (Minimal report without datapoints)
-----
{
    "name": "cpu-util",
    "datapointDuration": 15,
    "reportSize": 2
}
```

## Creating samples

The HTTP POST method is used to create samples for **Reports**.

Quick reference

```
POST http://localhost:8080/mws/rest/reports?api-version=3
```

## Create samples for report

URLs and parameters

```
POST http://localhost:8080/mws/rest/reports/<id>/samples?api-version=3
POST http://localhost:8080/mws/rest/reports/<name>/samples?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the report.
<b>name</b>	Yes	String	--	The name of the report.

 Only one of **id** or **name** are required.

See [Global URL parameters on page 1425](#) for available URL parameters.

Request body

To create samples for a report, simply send data and an optional timestamp to the URL above. The request body below shows all the fields that are available during sample creation. Note that the `data` field can contain arbitrary JSON.

JSON request body

```

{
  "timestamp": "2011-12-01 19:16:57 UTC",
  "agent": "my agent",
  "data": {
    "cpu1": 2.3,
    "cpu2": 1.2,
    "cpu3": 0.0,
    "cpu4": 12.1
  }
}

```

### Sample response

```
{"messages": ["1 sample(s) created for report cpu-util"]}
```

## Deleting reports

The HTTP DELETE method is used to delete **Reports**.

### Quick reference

```

DELETE http://localhost:8080/mws/rest/reports/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/reports/<name>?api-version=3

```

## Delete report

### URLs and parameters

```

DELETE http://localhost:8080/mws/rest/reports/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/reports/<name>?api-version=3

```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the report.
<b>name</b>	Yes	String	--	The name of the report.

 Only one of **id** or **name** are required.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

JSON response

```
{"messages": ["Report cpu-util deleted"]}
```

Related topics

- [Fields: Reports](#) on page 2261
- [Resources introduction](#) on page 1446

## Reservations

This section describes behavior of the **Reservations** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Reservations](#) reference contains the type and description of all fields in the **Reservations** object. It also contains details regarding which fields are valid during PUT and POST actions.

Supported methods

Resource	GET	PUT	POST	DELETE
<b>/rest/reservations</b>	<a href="#">Get all reservations</a>	--	<a href="#">Create reservation</a>	--
<b>/rest/reservations/&lt;id&gt;</b>	<a href="#">Get single reservation</a>	<a href="#">Modify reservation</a>	--	<a href="#">Release reservation</a>

This topic contains these sections:

- [Getting reservations](#) on page 1646
  - [Get all reservations](#) on page 1647
  - [Get single reservation](#) on page 1647
- [Creating reservations](#) on page 1650
  - [Create reservation](#) on page 1650
- [Modifying reservations](#) on page 1652
  - [Modify reservation](#) on page 1652
- [Releasing reservations](#) on page 1653
  - [Release reservation](#) on page 1653

### Getting reservations

The HTTP GET method is used to retrieve **Reservation** information. Queries for all objects and a single object are available.

Quick reference

```
GET http://localhost:8080/mws/rest/reservations/<id>?api-version=3
```

Restrictions

Only admin or user reservations are returned with this call.

**Get all reservations**

URLs and parameters

```
GET http://localhost:8080/mws/rest/reservations?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
GET http://localhost:8080/mws/rest/reservations?api-version=3&fields=id
-----
{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {"id": "system.1"},
    {"id": "system.2"},
    {"id": "system.3"}
  ]
}
```

**Get single reservation**

URLs and parameters

```
GET http://localhost:8080/mws/rest/reservations/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

JSON response

```

{
  "accountingAccount": "",
  "accountingGroup": "",
  "accountingQOS": "",
  "accountingUser": "root",
  "aclRules": [ {
    "affinity": "NEUTRAL",
    "comparator": "LEXIGRAPHIC_EQUAL",
    "type": "RESERVATION_ID",
    "value": "system.43"
  } ],
  "allocatedNodeCount": 1,
  "allocatedProcessorCount": 8,
  "allocatedTaskCount": 1,
  "allocatedNodes": [
    { "id": "node001" }
  ],
  "comments": "",
  "creationDate": null,
  "duration": 200000000,
  "endDate": "2018-03-17 16:49:10 UTC",
  "excludeJobs": [
    "job1",
    "job2"
  ],
  "expireDate": null,
  "flags": [
    "REQFULL",
    "ISACTIVE",
    "ISCLOSED"
  ],
  "globalId": "",
  "hostListExpression": "",
  "id": "system.43",
  "idPrefix": "",
  "isActive": true,
  "isTracked": false,
  "label": "",
  "maxTasks": 0,
  "messages": [],
  "owner": {
    "name": "adaptive",
    "type": "USER"
  },
  "partitionId": "switchB",
  "profile": "",
  "requirements": {
    "architecture": "",
    "featureList": [
      "feature1",
      "feature2"
    ],
    "featureMode": "",
    "memory": 0,
    "nodeCount": 0,
    "nodeIds": ["node001:1"],
    "os": "",
    "taskCount": 1
  },
}

```

```

"reservationGroup": "",
"resources": {"PROCS": 0},
"startDate": "2011-11-14 20:15:50 UTC",
"statistics": {
  "caps": 0,
  "cips": 2659.52,
  "taps": 0,
  "tips": 0
},
"subType": "Other",
"taskCount": 0,
"trigger": null,
"triggerIds": [],
"uniqueIndex": "",
"variables": {}
}

```

## Creating reservations

The HTTP POST method is used to create **Reservations**.

### Quick reference

```
POST http://localhost:8080/mws/rest/reservations?api-version=3
```

### Create reservation

#### URLs and parameters

```
POST http://localhost:8080/mws/rest/reservations?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Request body

The request body below shows all the fields that are available when creating a **Reservation**, along with some sample values.

JSON request body

```
{
  "accountingAccount": "",
  "accountingGroup": "",
  "accountingQOS": "",
  "accountingUser": "root",
  "aclRules": [ {
    "affinity": "POSITIVE",
    "comparator": "LEXIGRAPHIC_EQUAL",
    "type": "GROUP",
    "value": "staff"
  } ],
  "comments": "",
  "duration": 200000000,
  "endDate": "2018-03-17 16:49:10 UTC",
  "excludeJobs": [
    "job1",
    "job2"
  ],
  "flags": [
    "SPACEFLEX",
    "ACLOVERLAP",
    "SINGLEUSE"
  ],
  "hostListExpression": "",
  "idPrefix": "",
  "label": "myreservation",
  "owner": {
    "name": "adaptive",
    "type": "USER"
  },
  "partitionId": "",
  "profile": "",
  "requirements": {
    "architecture": "",
    "featureList": [
      "feature1",
      "feature2"
    ],
    "memory": 0,
    "os": "",
    "taskCount": 1
  },
  "reservationGroup": "",
  "resources": {
    "PROCS": 2,
    "MEM": 1024,
    "DISK": 1024,
    "SWAP": 1024,
    "other1": 17,
    "other2": 42
  },
  "startDate": "2011-11-14 20:15:50 UTC",
  "subType": "Other",
  "trigger": {
    "eventType": "START",
    "actionType": "EXEC",
    "action": "date"
  },
  "variables": {
```

```

    "var1": "val1",
    "var2": "val2"
  }
}

```

This example is to create a reservation if no conflicting reservations are found. (This is the equivalent to `mrsvctl -c -h node01 -E`.)

```

JSON request body
-----
{
  "flags": [
    "DEDICATEDRESOURCE"
  ],
  "hostListExpression": "node01"
}

```

Sample response

```

JSON Response for successful POST
-----
{"id": "system.44"}

```

## Modifying reservations

The HTTP PUT method is used to modify **Reservations**.

Quick reference

```

PUT http://localhost:8080/mws/rest/reservations/<id>?api-version=3&change-
mode=<add|remove|set>

```

### Modify reservation

URLs and parameters

```

PUT http://localhost:8080/mws/rest/reservations/<id>?api-version=3&change-
mode=<add|remove|set>

```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.
<b>change-mode</b>	Yes	String	add remove set	<p>If <b>add</b>, add the given variables to the variables that already exist.</p> <p>If <b>remove</b>, delete the given variables from the variables that already exist.</p> <p>If <b>set</b>, replace all existing variables with the given variables.</p>

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

The request body below shows all the fields that are available when modifying a **Reservation**, along with some sample values.

JSON request body for reservation modify

```
{
  "variables": {
    "var1": "val1",
    "var2": "val2"
  }
}
```

### Sample response

**i** This message may not match the message returned from Moab exactly, but is given as an example of the structure of the response.

JSON response

```
{"messages":["reservation 'system.43' attribute 'Variable' changed."]}
```

### Restrictions

You can change the ACL Rules on a reservation, but not using this resource. See [Create or update ACL on page 1449](#).

## Releasing reservations

The HTTP DELETE method is used to release **Reservations**.

### Quick reference

```
DELETE http://localhost:8080/mws/rest/reservations/<id>?api-version=3
```

## Release reservation

### URLs and parameters

```
DELETE http://localhost:8080/mws/rest/reservations/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

```
JSON Response for successful DELETE
{}

```

Related topics

- [Fields: Reservations on page 2270](#)
- [Resources introduction on page 1446](#)

## Resource types

This section describes behavior of the **Resource Type** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The [Fields: Resource Types](#) reference contains the type and description of all fields in the **Resource Type** object.

Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/resource-types</a>	<a href="#">Get all resource types</a>	--	--	--

This topic contains these sections:

- [Getting resource types on page 1654](#)
  - [Get all resource types on page 1654](#)

### Getting resource types

The HTTP GET method is used to retrieve **Resource Type** information.

Quick reference

```
GET http://localhost:8080/mws/rest/resource-types?api-version=3

```

### Get all resource types

URLs and parameters

```
GET http://localhost:8080/mws/rest/resource-types?api-version=3

```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/resource-types?api-version=3&fields=id
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [
    {"id": "throttle_migrate"}
  ]
}
```

### Related topics

- [Fields: Resource Types on page 2330](#)
- [Resources introduction on page 1446](#)

## Roles

This section describes behavior of the **Role** resource in Moab Web Services. The role resource is used to control access to MWS resources based on the proxy-user. Each role is attached to a principal and contains a list of proxy-user permissions that the group can use in MWS. This section describes the URLs, request bodies, and responses delivered to and from MWS.



The [Fields: Roles](#) reference section contains the type and description of all fields in the **Role** object. It also contains details regarding which fields are valid during PUT and POST actions.

### Supported methods

Resource	GET	PUT	POST	DELETE
<a href="#">/rest/roles</a>	<a href="#">Get all roles</a>	--	<a href="#">Create single role</a>	--
<a href="#">/rest/roles/&lt;id&gt;</a>	<a href="#">Get single role</a>	<a href="#">Modify single role</a>	--	<a href="#">Deleting roles</a>
<a href="#">/rest/roles/&lt;name&gt;</a>	<a href="#">Get single role</a>	<a href="#">Modify single role</a>	--	<a href="#">Delete single role</a>

This topic contains these sections:

- [Getting roles on page 1656](#)
  - [Get all roles on page 1656](#)
  - [Get single role on page 1657](#)
- [Creating roles on page 1658](#)
  - [Create single role on page 1659](#)

- [Modifying roles on page 1660](#)
  - [Modify single role on page 1660](#)
- [Deleting roles on page 1661](#)
  - [Delete single role on page 1661](#)

## Getting roles

The HTTP GET method is used to retrieve **Role** information. You can query all objects or a single object.

### Quick reference

```
GET http://localhost:8080/mws/rest/roles?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
GET http://localhost:8080/mws/rest/roles/<id>?api-version=3
GET http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

## Get all roles

### URLs and parameters

```
GET http://localhost:8080/mws/rest/roles?api-version=3[&query={"field":"value"}&sort={"field":<1|-1>}]
```

Parameter	Required	Type	Valid values	Description	Example
<b>query</b>	No	JSON	--	Queries for specific results. It is possible to query roles by one or more fields based on <a href="#">MongoDB query syntax</a> .	query={"name":"Acme-User-Role"}
<b>sort</b>	No	JSON	--	Sort the results. Use 1 for ascending and -1 for descending.	sort={"name":-1}

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
GET http://localhost:8080/mws/rest/roles?api-version=3&fields=id,name
-----
{
  "totalCount": 1,
  "resultCount": 1,
  "results": [ {
    "id": "4fa197e68ca30fc605dd1cf0",
    "name": "Acme-User-Role"
  } ]
}
```

### Sorting and querying

See the sorting and querying sections of [Global URL parameters on page 1425](#).

## Get single role

### URLs and parameters

```
GET http://localhost:8080/mws/rest/roles/<id>?api-version=3
GET http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Role.
<b>name</b>	Yes	String	--	The name of the Role.



You must specify either **id** or **name**, but you do not have to specify both.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```

GET http://localhost:8080/mws/rest/roles/Acme-User-Role?api-version=3
-----
{
  "description" : "This is a role for normal users in the Acme BU Group.",
  "id" : "5022e695e4b073f54e47c28d",
  "name" : "Acme-User-Role",
  "permissions" : [ {
    "action" : "create",
    "description" : "The permission to create all charts.",
    "id" : "5022e695e4b073f54e47c28e",
    "label" : "Create Chart",
    "resource" : "chart",
    "resourceFilter" : null,
    "type" : "custom",
    "scope" : "GLOBAL",
    "version" : 0
  }, {
    "action" : "read",
    "description" : "The permission to view all charts.",
    "id" : "5022e695e4b073f54e47c28f",
    "label" : "View Chart",
    "resource" : "chart",
    "resourceFilter" : null,
    "type" : "custom",
    "scope" : "GLOBAL",
    "version" : 0
  }, {
    "action" : "update",
    "description" : "The permission to modify the africa chart.",
    "id" : "5022e695e4b073f54e47c290",
    "label" : "Modify Africa Chart",
    "resource" : "chart",
    "resourceFilter" : {
      "name" : "africa"
    },
    "type" : "custom",
    "scope" : "GLOBAL",
    "version" : 0
  }, {
    "action" : "read",
    "description" : "The permissions to view John's services.",
    "id" : "5022e695e4b073f54e47c291",
    "label" : "Read John's services",
    "resource" : "services",
    "resourceFilter" : {
      "user": "john"
    },
    "type" : "api",
    "scope" : "GLOBAL",
    "version" : 0
  } ],
  "version" : 2
}

```

## Creating roles

The HTTP POST method is used to submit **Roles**.

### Quick reference

```
POST http://localhost:8080/mws/rest/roles?api-version=3
```

## Create single role

### URLs and parameters

```
POST http://localhost:8080/mws/rest/roles?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

### Request body

**i** The **name** field is required and must contain only letters, digits, periods, dashes, and underscores.

The following is an example of a request body to create a role:

```
POST http://localhost:8080/mws/rest/roles?api-version=3
-----
{
  "name" : "Acme-User-Role",
  "description" : "This is a role for normal users in the Acme BU Group.",
  "permissions" :
  [
    {
      "id" : "4fa197e68ca30fc605dd1cf0"
    },
    {
      "id" : "4fa197e68ca30fc605dd1df2"
    }
  ]
}
```

### Sample response

If the request was successful, the response body is the new role that was created, exactly as shown in [Get single role](#). On failure, the response is an error message.

### Samples

The **permissions** field only expects an array of permission IDs, as shown in the following example:

```
Example payload of role with 2 permissions
-----
{
  "name" : "Acme-User-Role",
  "description" : "This is a role for normal users in the Acme BU Group.",
  "permissions" :
  [
    {
      "id" : "4fa197e68ca30fc605dd1cf0"
    }
  ]
}
```

## Modifying roles

The HTTP PUT method is used to modify **Roles**.

### Quick reference

```
PUT http://localhost:8080/mws/rest/roles/<id>?api-version=3
PUT http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

## Modify single role

### URLs and parameters

```
PUT http://localhost:8080/mws/rest/roles/<id>?api-version=3
PUT http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Role.
<b>name</b>	Yes	String	--	The name of the Role.  <div style="border: 1px solid #0070C0; padding: 5px; border-radius: 5px;">  The <b>name</b> field must contain only letters, digits, periods, dashes, and underscores.                 </div>
<b>change-mode</b>	No	String	add remove set (default)	If <b>add</b> , adds the given permissions to the permissions that already exist. If <b>remove</b> , deletes the given permissions from the permissions that already exist. If <b>set</b> , adds the given permissions and deletes the permissions that already exist.

 You must specify either **id** or **name**, but you do not have to specify both.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Example request

```
PUT http://localhost/mws/rest/role/Acme-User-Role?change-mode=add?api-version=3
{
  "permissions": [{"id": "4fa197e68ca30fc605dd1cf0"} ]
}
```

### Sample response

If the request was successful, the response body is the modified role as shown in [Get single role](#). On failure, the response is an error message.

## Deleting roles

The HTTP DELETE method is used to delete **Roles**.

### Quick reference

```
DELETE http://localhost:8080/mws/rest/roles/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

### Delete single role

#### URLs and parameters

```
DELETE http://localhost:8080/mws/rest/roles/<id>?api-version=3
DELETE http://localhost:8080/mws/rest/roles/<name>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the Role.
<b>name</b>	Yes	String	--	The name of the Role.



You must specify either **id** or **name**, but you do not have to specify both.

See [Global URL parameters on page 1425](#) for available URL parameters.

### Sample response

```
JSON response
```

```
{}
```

### Related topics

- [Fields: Roles on page 2331](#)
- [Resources introduction on page 1446](#)

## Standing reservations

This section describes behavior of the **Standing Reservation** object in Moab Web Services. It contains the URLs, request bodies, and responses delivered to and from MWS.

**i** The **Fields: Standing Reservations** reference section contains the type and description of all fields in the **Standing Reservation** object. It also contains details regarding which fields are valid during PUT and POST actions.

### Supported methods

Resource	GET	PUT	POST	DELETE
<code>/rest/standing-reservations</code>	<a href="#">Get all standing reservations</a>	--	--	--
<code>/rest/standing-reservations/&lt;id&gt;</code>	<a href="#">Get single standing reservation</a>	--	--	--

This topic contains these sections:

- [Getting standing reservations on page 1662](#)
  - [Get all standing reservations on page 1662](#)
  - [Get single standing reservation on page 1663](#)

## Getting standing reservations

The HTTP GET method is used to retrieve **Standing Reservation** information. Queries for all objects and a single object are available.

### Quick reference

```
GET http://localhost:8080/mws/rest/standing-reservations/<id>?api-version=3
```

### Get all standing reservations

#### URLs and parameters

```
GET http://localhost:8080/mws/rest/standing-reservations?api-version=3
```

See [Global URL parameters on page 1425](#) for available URL parameters.

#### Sample response

```
GET http://localhost:8080/mws/rest/standing-reservations?api-version=3&fields=id
-----
{
  "totalCount": 3,
  "resultCount": 3,
  "results": [
    {"id": "sr1"},
    {"id": "sr2"},
    {"id": "sr3"}
  ]
}
```

## Get single standing reservation

### URLs and parameters

```
GET http://localhost:8080/mws/rest/standing-reservations/<id>?api-version=3
```

Parameter	Required	Type	Valid values	Description
<b>id</b>	Yes	String	--	The unique identifier of the object.

See [Global URL parameters on page 1425](#) for available URL parameters.

Sample response

JSON response

```

{
  "access": "DEDICATED",
  "accounts": ["account1"],
  "aclRules": [ {
    "affinity": "POSITIVE",
    "comparator": "EQUAL",
    "type": "USER",
    "value": "adaptive",
  } ],
  "chargeAccount": "account2",
  "chargeUser": "user2",
  "classes": ["class1"],
  "clusters": ["cluster1"],
  "comment": "comment",
  "days": ["Monday"],
  "depth": 2,
  "disabled": false,
  "endOffset": 86415,
  "flags": ["ALLOWJOB OVERLAP"],
  "groups": ["group1"],
  "hosts": ["host1"],
  "id": "fast",
  "jobAttributes": ["TEMPLATESAPPLIED"],
  "maxJob": 2,
  "maxTime": 0,
  "messages": ["message1"],
  "nodeFeatures": ["feature1"],
  "os": "Ubuntu 10.04.3",
  "owner": {
    "name": "root",
    "type": "USER"
  },
  "partition": "ALL",
  "period": "DAY",
  "procLimit": {
    "qualifier": "<=",
    "value": 5
  },
  "psLimit": {
    "qualifier": "<=",
    "value": 60
  },
  "qoses": ["qos1"],
  "reservationAccessList": [],
  "reservationGroup": "group2",
  "resources": {
    "PROCS": -1,
    "tapes": 1
  },
  "rollbackOffset": 43200,
  "startOffset": 347040,
  "taskCount": 0,
  "tasksPerNode": 0,
  "timeLimit": -1,
  "triggers": [],
  "type": "type1",
  "users": ["user1"]
}

```

## Related topics

- [Fields: Standing Reservations on page 2343](#)
- [Resources introduction on page 1446](#)

# Reporting framework

## Overview of reporting framework

The reporting framework is a set of tools to make time-based reports from numerical data. The following sections will (1) provide an overview of the framework and the concepts related to it, and (2) work through an example report (CPU Utilization) with details regarding which web services to use and with what data.

The REST API reference is located in the Report resource section (see [Reports on page 1636](#)).

## Concepts

The reporting framework uses 3 core concepts: reports, datapoints, and samples.

- Reports (see [Fields: Reports on page 2261](#)): A report is a time-based view of numerical data.
- Report Datapoints (see [Fields: Report Datapoints on page 2259](#)): A datapoint is a consolidated set of data for a certain time period.
- Report Samples (see [Fields: Report Samples on page 2341](#)): A sample is a snapshot of a certain set of data at a particular point in time.

To illustrate, consider the memory utilization of a virtual machine: at any given point in time, you can get the memory utilization by using your operating system's performance utilities (top for Linux, Task Manager for Windows):

2400/12040MB

By recording the memory utilization and time constantly for 1 minute, you could gather the following data:

Time	Memory utilization
3:53:55 PM	2400/12040 MB
3:54:13 PM	2410/12040 MB
3:54:27 PM	2406/12040 MB
3:54:39 PM	2402/12040 MB

Time	Memory utilization
3:54:50 PM	2409/12040 MB

Each of the rows in the table above represent a **sample** of data. By averaging the rows we can consolidate them into one or more **datapoints**:

Start time	End time	Memory utilization
3:53:30 PM	3:54:00 PM	2400/12040 MB
3:54:00 PM	3:54:30 PM	2408/12040 MB
3:54:30 PM	3:55:00 PM	2406/12040 MB

**i** Note that each datapoint covers exactly the same amount of time, and averages all samples within that period of time.

A **report**, then, is simply a list of datapoints with some additional configuration information:

Field	Value
<b>Name</b>	Memory Utilization Report
<b>Datapoint Duration</b>	30 seconds
<b>Report Size</b>	3 datapoints

Datapoints:

Start time	End time	Memory utilization
3:53:30 PM	3:54:00 PM	2400/12040 MB
3:54:00 PM	3:54:30 PM	2408/12040 MB
3:54:30 PM	3:55:00 PM	2406/12040 MB

## Capabilities

While storing simple information like memory utilization is nice, the reporting framework is built to automatically handle much more complex information.

### Consolidating Samples

Samples are JSON documents which are pushed into the report using the Samples API (see [Creating samples on page 1644](#)). Samples are then stored until the consolidation operation creates a datapoint out of them. The table below shows how different data types are handled in this operation:

Type	Consolidation function handling
<b>Numbers</b>	Numerical data is averaged.
<b>Strings</b>	Strings are aggregated into an array.
<b>Objects</b>	The consolidation function recursively consolidates sub-objects.
<b>Lists</b>	Lists are combined into a single flat list containing all elements.
<b>Mixed</b>	If samples have different types of data for the same field, the values are aggregated into an array.
<b>Null</b>	These values will be ignored unless all values for a sample field are set to null, resulting in a null result.

**i** If the mixed data types contains at least one number, it will be treated as numerical data. The non-numerical data will be ignored and the result will be averaged.

Below is an example of how the consolidation function works:

- Samples:

Time	NumberEx	StringEx	ListEx	MixedEx	MixedNumberEx
3:53:55 PM	2400	"str1"	["elem1"]	"str1"	"str1"
3:54:13 PM	2410	"str2"	["elem2", "elem3"]	["elem1"]	["elem1"]
3:54:27 PM	2405	"str3"	["elem4"]	null	5

- Resulting Datapoint after consolidation:

Time	NumberEx	StringEx	ListEx	MixedEx	MixedNumberEx
3:55:00 PM	2405	["str1", "str2", "str3"]	["elem1", "elem2", "elem3", "elem4"]	["str1", "elem1"]	5

### Minimum number of samples

If your dataset is highly variable (i.e. values contained in samples are not very close together), converting a single sample into a datapoint may provide misleading information. It may be better to have a datapoint with an "Unknown" value. This can be accomplished by setting the minimum number of samples for a datapoint in the report.

The `minimumSampleSize` field in the Reports reference section (see [Reports on page 1636](#)) explains that if the specified size of samples is not met when the consolidation function is performed, the datapoint is considered "null" and no data is available for it. When this occurs, the sample data is discarded and the `data` field of the datapoint is set to "null".

For information on how to set this option, see the REST API Report Resource section (see [Reports on page 1636](#)).

### Report size

Reports have a predetermined number of datapoints, or size, which sets a limit on the amount of data that can be stored. After the report size has been reached, as newly created datapoints are pushed into the report, the oldest datapoints will automatically be deleted. This is to aid in managing the storage capacity of the server hosting MWS.



On report creation, a Mongo collection will be initialized that is the configured report document size multiplied by the report size. Be careful in setting a large report size or report document size as this may quickly allocate the entire disk. See the `reportDocumentSize` and `reportSize` fields in [Fields: Reports on page 2261](#) for more information.

### Related topics

- [Example report \(CPU Utilization\) on page 1668](#)

## Example report (CPU Utilization)

To understand how the behavior and usage of the reporting framework, a sample report covering CPU Utilization will be shown in this section. It will not cover how to gather or display data for reports, but will cover some basic operations that are available with Moab Web Services to facilitate reporting.

### Creating a report

Before any data is sent to Moab Web Services, a report must first be created. A JSON request body with a HTTP method of POST must be used to do this.

```

POST /rest/reports
-----
{
  "name": "cpu-util",
  "description": "An example report for cpu utilization",
  "consolidationFunction": "average",
  "datapointDuration": 600,
  "reportSize": 288
}

```

This will result in a report being created which can then be retrieved by sending a GET request to `/rest/reports/cpu-util`. The `datapointDuration` of 600 signifies that the datapoint consolidation should occur once every 10 minutes, while the `reportSize` (i.e. number of the datapoints) shows that the report will retain up to 2 days worth of the latest datapoints.

```

GET /rest/reports/cpu-util
-----
{
  "consolidationFunction": "average",
  "datapointDuration": 600,
  "datapoints": [],
  "description": "An example report for cpu utilization",
  "id": "aef6f6a3a0bz7bf6449537c9d",
  "keepSamples": false,
  "minimumSampleSize": 1,
  "name": "cpu-util",
  "reportSize": 288,
  "version": 0
}

```

(Note that an ID has been automatically generated and that no datapoints are associated with the report.)

## Adding samples

Until samples are added and associated with the report, datapoint consolidation will generate datapoints with a `data` field equal to `null`. Once samples are added, however, they will be averaged and inserted into the next datapoint.

Create samples for the `cpu-util` by sending a POST request as follows:

```
POST /rest/reports/cpu-util/samples
```

```
[
  {
    "agent": "cpu-monitor",
    "timestamp": "2012-01-01 12:00:00 UTC",
    "data": {
      "minutes1": 0.5,
      "minutes5": 0,
      "minutes15": 0
    }
  },
  {
    "agent": "cpu-monitor",
    "timestamp": "2012-01-01 12:01:00 UTC",
    "data": {
      "minutes1": 1,
      "minutes5": 0.5,
      "minutes15": 0.05
    }
  },
  {
    "agent": "cpu-monitor",
    "timestamp": "2012-01-01 12:02:00 UTC",
    "data": {
      "minutes1": 1,
      "minutes5": 0.5,
      "minutes15": 0.1
    }
  },
  {
    "agent": "cpu-monitor",
    "timestamp": "2012-01-01 12:03:00 UTC",
    "data": {
      "minutes1": 0.75,
      "minutes5": 1,
      "minutes15": 0.25
    }
  },
  {
    "agent": "cpu-monitor",
    "timestamp": "2012-01-01 12:04:00 UTC",
    "data": {
      "minutes1": 0,
      "minutes5": 1,
      "minutes15": 0.85
    }
  }
]
```

This sample data contains average load for the last 1, 5, and 15 minute intervals. The samples were recorded at one-minute intervals starting at noon on January 1st, 2012.

## Consolidating data

A consolidation function must run to generate datapoints from the given samples. This scheduled consolidation will occur at intervals of `datapointDuration` seconds. For each field in the `data` object in samples, all values will be averaged. If non-numeric values are included, the following strategies will be followed:

1. All fields which contain a single numeric value in any included sample will be averaged and the non-numeric or null values will be ignored.
2. All fields which contain a list will be consolidated into a single, flat list.
3. All fields which contain only non-numeric or null values will be consolidated into a single, flat list.

If no historical datapoints are provided in the creation of a report as in this example, the next consolidation will be scheduled for the current time plus the `datapointDuration`. In this example, the scheduled consolidation is at 10 minutes from the creation date. If historical datapoints are included in the report creation, the latest datapoint's `endDate` plus the `datapointDuration` will be used as the scheduled time. If this date was in the past, the next scheduled consolidation will occur at the appropriate interval from the last `endDate`.

## Retrieving report data

To retrieve the consolidated datapoints, simply perform a GET request on the report once again. Alternatively, the GET for a report's datapoints (see [Get datapoints for single report on page 1640](#)) may be used.

```
GET /rest/reports/cpu-util
-----
{
  "consolidationFunction": "average",
  "datapointDuration": 600,
  "datapoints": [
    {
      "firstSampleDate": null,
      "lastSampleDate": null,
      "data": null,
      "startDate": "2012-01-01 11:49:00 UTC",
      "endDate": "2012-01-01 11:59:00 UTC"
    },
    {
      "firstSampleDate": "2012-01-01 12:00:00 UTC",
      "lastSampleDate": "2012-01-01 12:04:00 UTC",
      "data": {
        "minutes1": 0.65,
        "minutes15": 0.25,
        "minutes5": 0.6
      },
      "startDate": "2012-01-01 11:59:00 UTC",
      "endDate": "2012-01-01 12:09:00 UTC"
    }
  ],
  "description": "An example report for cpu utilization",
  "id": "aef6f6a3a0bz7bf6449537c9d",
  "keepSamples": false,
  "minimumSampleSize": 1,
  "name": "cpu-util",
  "reportSize": 288,
  "version": 0
}
```

Note that of the two datapoints above, only the second actually contains data, while the other is set to null. Only samples lying within the datapoint's duration, or from the `startDate` to the `endDate`, are included in the consolidation. Therefore the first datapoint, which covered the 10 minute period just

before the samples' recorded timestamps, contained no data. The second, which covers the 10 minute period matching that of the samples, contains the averaged sample data. This data could be used to display consolidated report data in a custom interface.

## Possible configurations

Configuration options may be changed to affect the process of report generation. These are documented in [Fields: Reports on page 2261](#) and [Fields: Report Samples on page 2341](#).

Related topics

- [Overview of reporting framework on page 1665](#)

# Plugins

## About Moab Web Services plugins

This chapter describes MWS plugins, their use, and their creation in Moab Workload Manager. The sections in this chapter provide you with the following information:

- An introduction to the concept of MWS plugins (see [Plugin introduction on page 1673](#)).
- A description of the plugin lifecycle (see [Lifecycle states on page 1675](#)).
- How plugins are driven by events ([Handling events on page 1711](#)).
- How to expose web services from a plugin ([Exposing web services on page 1696](#)).
- How plugin utility services may be used ([Utility services on page 1677](#)).
- How data report collisions between plugins are consolidated ([Data consolidation on page 1677](#)).
- How calls from Moab are routed to MWS plugins ([Routing on page 1679](#)).

It contains the following sections:

- [Plugin overview on page 1673](#)
- [Plugin developer's guide on page 1679](#)
- [Plugin type management on page 1737](#)
- [Plugin management on page 1742](#)
- [Plugin services on page 1747](#)

Related topics

- [Configuring Moab Web Services on page 1395](#)

## Plugin overview

This section provides an overview of the plugin layer in web services. It contains these topics:

- [Plugin introduction on page 1673](#)
- [Lifecycle states on page 1675](#)
- [Events on page 1676](#)
- [Custom web services on page 1676](#)
- [Utility services on page 1677](#)
- [Data consolidation on page 1677](#)
- [Routing on page 1679](#)

Related topics

- [About Moab Web Services plugins on page 1672](#)

## Plugin introduction

Moab Web Services plugins provide a highly extensible interface to interact with Moab, MWS, and external resources. Plugins can perform some of the same functions as Moab resource managers (RMs), while also providing many other features not available to RMs. This section will discuss the main features of plugins, some basic terminology, and how MWS plugins can interact with Moab.

### Features

Plugins can:

- Be created, modified, and deleted without restarting Moab Workload Manager or MWS.
- Be defined in Groovy and uploaded to MWS without restarting.
- Have individual data storage space and configuration.
- Access MWS configuration and RESTful web services.
- Log to a standard location configured in MWS.
- Be polled at a regular interval (configured on a per-plugin basis).
- Be informed of important system events.
- Be individually stopped, started, paused, and resumed.
- Expose secured and unsecured custom web services for external use.
- Be manipulated via a full RESTful API (for more information, see [Resources introduction on page 1446](#)).
- Be manipulated via a full user interface in a web browser.

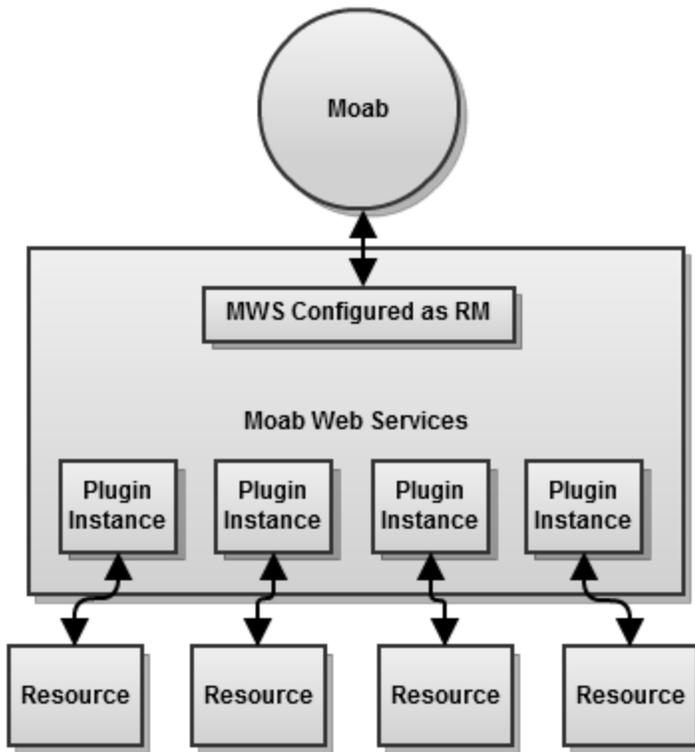
### Terminology

There are two distinct terms in the plugin layer: [plugin types](#) and [plugins \(instances\)](#).

Term	Description
<b>plugin types</b>	<p>Plugin types can be considered plugin templates with built-in logic. In object-oriented programming languages, this relates to the concept of a class. They possess certain abilities, or methods, that can be called by Moab Web Services to query or update information about certain resources. They also can define methods which will be exposed to external clients as web services. They do not contain any configuration or current data, but they are often tied to a <i>type</i> of component, such as components that communicate with Moab's WIKI Protocol, or those that are built on a certain product.</p> <p>They can define several types of methods:</p> <ul style="list-style-type: none"> <li>• <b>Instance methods</b> that return information about the current plugin, such as <code>getState</code>. (While these are defined in the plugin type, the plugin type itself does not have a state.)</li> <li>• The <b>poll event method</b> that is called at a configured interval.</li> <li>• <b>Lifecycle event methods</b> of plugins created from the plugin type, such as <code>beforeStart</code> and <code>afterStart</code>.</li> <li>• <b>RM event methods</b> that are called by Moab when certain events occur.</li> <li>• <b>Web service methods</b> that expose custom functionality as public web services.</li> </ul> <p>Some examples of plugin types include the Native and vCenter plugin types.</p>
<b>plugins (instances)</b>	<p>Plugins (also called plugin instances) are created from plugin types. They contain current data or configuration and use the plugin type methods to interact with resources.</p>

#### Interactions with Moab as a resource manager

The plugin layer in MWS is integrated with Moab Workload Manager via the Native Resource Manager (RM) interface. When utilizing plugins, MWS is configured as a RM in Moab, as explained in the next section. Events from Moab are pushed through the RM interface to MWS, which is then pushed to each plugin in turn. The relationship between MWS, Moab, and plugins is shown in the following image:



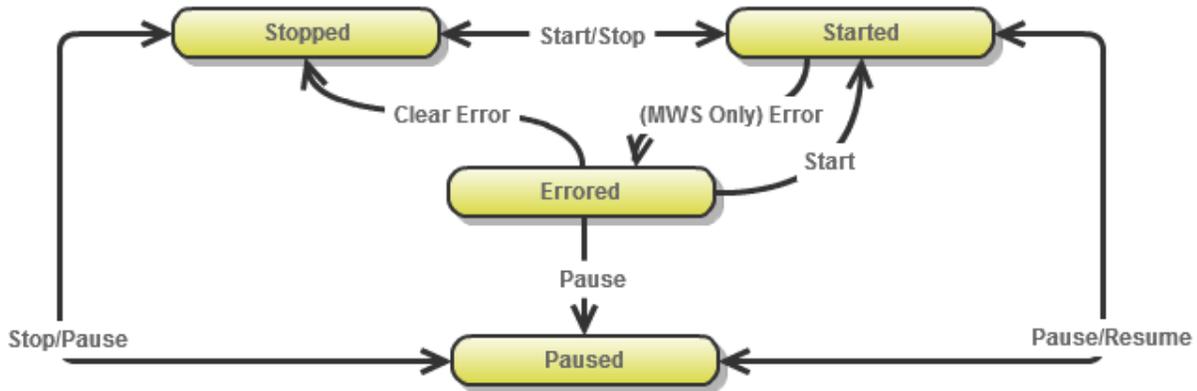
For more information, see [Data consolidation](#) on page 1677 and [Reporting state data](#) on page 1698.

Related topics

- [About Moab Web Services plugins](#) on page 1672

## Lifecycle states

During the course of a plugin's use, the state of the plugin may change many times. Plugins have four possible states: **Stopped**, **Started**, **Paused**, and **Errored**. For the descriptions of each state, see the [Fields: Plugins](#) reference section. The flow of a plugin through the states is shown in the following image:



**i** You can see [Handling events](#) on page 1711 for information about the events that occur during lifecycle state changes.

Related topics

- [Plugin introduction](#) on page 1673

Events

Plugins use an event-based model, meaning that methods are called on the plugin when certain criteria are met or situations arise. Events currently exist for polling, lifecycle state changes, and RM events from Moab. For more information, see [Handling events](#) on page 1711.

Related topics

- [Handling events](#) on page 1711
- [Plugin introduction](#) on page 1673

Custom web services

Although the events interface typically serves most cases, there are some instances where an event is not supported that is desired. This is especially true when an external resource is the source of the event. To address these issues, plugins can expose custom web services to external resources. These web services may be named freely and do anything they wish within the plugin framework.

For example, suppose a resource needs to notify a plugin that provisioning of a virtual machine has been completed. Instead of having the plugin poll the resource to verify that the provisioning was finished, the plugin could expose a custom web service to handle notification from the resource itself.

```

Sample custom web service
-----

def vmProvisionFinished(Map params) {
    // Handle event
    return [messages:["Event successfully processed"]]
}

```

Additionally, plugin types may define web services which are unsecured, meaning that a user or application account is not required to access it. A full explanation of the syntax and creation of custom secured and unsecured web services may be seen on [Exposing web services on page 1696](#).

For information how resources can access plugin web services, see [Accessing Plugin Web Services on page 1605](#).

Related topics

- [Plugin introduction on page 1673](#)

## Utility services

Several features of plugins are only available by utilizing bundled services. These include:

- Accessing the individual datastore (see [Individual datastore on page 1694](#)).
- Reporting state data to Moab through the Resource Manager interface (see [Reporting state data on page 1698](#)).
- Manipulating other plugins and controlling their lifecycle (see [Controlling lifecycle on page 1701](#)).
- Accessing REST resources from Moab Web Services ([Accessing MWS REST resources on page 1702](#)).

It may also be necessary or desired to create additional utility services when creating new plugin types. The easiest way to do this is to create a utility service which is called by convention a translator (see [Using translators on page 1716](#)), because it can typically "translate" from a specific resource or API to data which can be used by the plugin type.

Finally, custom components (see [Registering custom components on page 1717](#)) may be used to fulfill use cases not covered by bundled services or custom translators.

Related topics

- [Plugin introduction on page 1673](#)

## Data consolidation

At times, plugins can report differing or even contradictory data for nodes, virtual machines, and jobs. This is called a data "collision". The act of resolving these collisions is called "Consolidation." Plugins also have the concept of "precedence," where the plugins with the lowest precedence value are considered more authoritative than the greater precedence values plugins. For example, a plugin with a precedence value of 1 has a higher precedence and is considered more authoritative than a plugin with a precedence value of 5. If no precedence is provided when creating plugins, the plugin is automatically assigned to the

lowest precedence, or 1 greater than the highest precedence value. The precedence value may not be less than 1.

When data from one plugin "collides" with another, the data from the highest precedence plugin will be considered the authoritative source for information. If multiple sets of data (reports) are provided by the same plugin, the latest set of data will take precedence. Additionally, MWS supports the concept of treating node and virtual machine data with state information **optimistically**, **pessimistically**, or **neither**. This is configured using the `plugins.stateConsolidationPolicy` configuration property in the MWS configuration file. If this property is set to `optimistic` and *any* plugin reports the state for a node or VM as "Up," the consolidated state will be "Up." Inversely, if the property is set to `pessimistic` and *any* plugin reports the state as "Down," the consolidated state will be "Down." If it is set to `null` (neither), consolidation will occur for the state field just as with any other field, with higher precedence and later reports being considered authoritative.

**i** When MWS is upgraded to a version that supports plugin precedence from an older version, existing plugins will not have the precedence field set. The administrator should assign precedence to each plugin manually through the API (see [Modifying plugins on page 1603](#)) or through the user interface (see [Modifying a plugin on page 1744](#)) to ensure that the consolidation will occur as expected. By default, data from a plugin without a precedence defaults to a precedence of 1, or the highest precedence.

Consolidation examples

Suppose two plugins exist, `pluginA` and `pluginB`. Plugin "A" has a precedence of 1, and plugin "B" has a precedence of 2, meaning that plugin "A" is more authoritative. These plugins both report data for a node with an ID of `node1`. However, each reports a different node power state. Plugin "A" reports the power as `ON`, while plugin "B" reports the power as `OFF`. The data collision that occurs due to these two contradictory reports is resolved by the precedence of the plugins. Since plugin A has a higher precedence (lower number), it is considered authoritative and the node will be reported as `ON`.

Now suppose that the plugins also report differing node state for `node1`. In this case, the node state would depend on the `plugins.stateConsolidationPolicy` property. The different combinations of report values compared to the state consolidation policy and the final reported state are shown in the table below.

Plugin "A" node state	Plugin "B" node state	State consolidation policy	Consolidated node state
ON	OFF	null (neither)	ON
OFF	ON	null (neither)	OFF
ON	OFF	optimistic	ON
OFF	ON	optimistic	ON

Plugin "A" node state	Plugin "B" node state	State consolidation policy	Consolidated node state
ON	OFF	pessimistic	OFF
OFF	ON	pessimistic	OFF

In general, it is recommended that no two plugins report the same resource or that they report different properties of the same resource. For example, if plugin "A" only modified the power state and plugin "B" only modified the available disk resource, these two plugins would work in harmony to provide a consistent view of the node resource.

For more information, see [Reporting state data on page 1698](#) and [Resource manager queries on page 1735](#).

Related topics

- [Plugin introduction on page 1673](#)

## Routing

 Plugin routing is currently in *Beta*. Interfaces may change significantly in future releases.

Because Moab Web Services is configured as a Resource Manager (RM) in Moab Workload Manager, events are sometimes triggered by Moab through the RM interface. These actions could be migrating a virtual machine, starting a job, submitting a job, modifying a node, and so forth. The decisions regarding which plugins are affected and notified is termed **routing**.

Currently all plugins receive all commands from Moab. This means that each plugin will receive the command to start a job if sent from Moab, even if that plugin does not handle the job. This means that plugins must ensure they handle actions or commands only for resources which they report or handle.

Related topics

- [Plugin introduction on page 1673](#)

## Plugin developer's guide

Plugin types comprise the methods by which Moab may communicate with resource managers or other external components. They define all operations that can be performed for a "type" or "class" of plugins, hence the name "plugin type."

Several plugin types are provided with Moab Web Services, but it is easy to create additional plugin types and add their functionality to web services. This involves using [Groovy](#), which is based on the [Java](#) programming language. This section describes the general guidelines and specifics of implementing new plugin types.

## API classes and interfaces

There are several packages and classes available to assist in creating plugin types. These can all be found in the [API documentation](#).

This section contains these topics:

- [Requirements](#) on page 1680
- [Dynamic methods](#) on page 1681
- [Logging](#) on page 1682
- [i18n messaging](#) on page 1683
- [Configuration](#) on page 1685
- [Configuration constraints](#) on page 1686
- [Individual datastore](#) on page 1694
- [Exposing web services](#) on page 1696
- [Reporting state data](#) on page 1698
- [Controlling lifecycle](#) on page 1701
- [Accessing MWS REST resources](#) on page 1702
- [Creating events and notifications](#) on page 1704
- [Handling events](#) on page 1711
- [Handling exceptions](#) on page 1713
- [Managing SSL connections](#) on page 1714
- [Utilizing services or custom "helper" classes](#) on page 1715
- [Packaging plugins](#) on page 1720
- [Example plugin types](#) on page 1729

Related topics

- [About Moab Web Services plugins](#) on page 1672

## Requirements

This section discusses the requirements to create a basic functional plugin. The `com.adaptc.mws.plugins` package contains the abstract class [AbstractPlugin](#) that should form the basis of any new plugin type. However, this class need not be extended to create a functional plugin type. Only two requirements must be fulfilled for this:

1. The class name must end in `Plugin`.
2. There must exist `id` field getter and setter methods:

```
* public String getId();  
* public void setId(String id);
```

The `id` field may be stored in whichever way desired as long as the getter and setter are available as shown above, but will most likely be implemented as follows:

```
class BasicPlugin {
    String id
}
```

In this case, `String id` will be expanded by the Groovy compiler to the full getter and setter method definitions given above. In other words, no explicit method definitions are actually needed. Note that the `BasicPlugin` shown above is able to be uploaded as a plugin type to MWS, but does not actually do anything.

It must also be noted that the `AbstractPlugin` class already implements an `id` field. Therefore, a plugin type that extends this class does not need to define the field as shown in the following example.

```
import com.adaptc.mws.plugins.AbstractPlugin

class BasicPlugin extends AbstractPlugin {
    // No ID field is needed since it exists in AbstractPlugin
}
```

#### Related topics

- [Plugin developer's guide on page 1679](#)

## Dynamic methods

 These methods are currently in *Beta*. Interfaces may change significantly in future releases.

Several methods are dynamically inserted onto each plugin. These methods do not need to be included in the plugin class, and will be overwritten if included. Additionally, a logger is inserted into each plugin as discussed in the next section. The inserted methods are shown below (full definitions can be found in [AbstractPlugin](#) and [AbstractPluginInfo](#)):

- `public void start() throws PluginStartException;` (Equivalent to the start method in the [Plugin control service on page 1750](#).)
- `public void stop() throws PluginStopException;` (Equivalent to the stop method in the [Plugin control service on page 1750](#).)
- `public Log getLog();` (See [Logging on page 1682](#).)
- `public ConfigObject getAppConfig();` (See [Configuration on page 1685](#).)
- `public String message(Map parameters);` (See [i18n messaging on page 1683](#).)
- `public String getPluginType();`
- `public PluginState getState();`
- `public Integer getPollInterval();`
- `public Boolean getAutoStart();`
- `public Map<String, Object> getConfig();` (See [Configuration on page 1685](#).)

Many of these methods are provided for convenience and are discussed in the linked pages or the following sections.

### Related topics

- [Plugin developer's guide on page 1679](#)

## Logging

Logging in plugin types uses the [Apache Commons Logging](#) and [log4j](#) libraries. Each plugin is injected with a method called `getLog` which can be used to access the configured logger. It returns an instance of [org.apache.commons.logging.Log](#). Examples of using the logger are shown below.

The logger may be used to register messages to the MWS log at several levels (in order of severity):

1. trace
2. debug
3. info
4. warn
5. error
6. fatal

Each of these levels is available as a method on the logger, for example:

```
public void poll() {
    getLog().debug("getLog() is equivalent to just using 'log' in Groovy")
    log.debug("This is a debug message and is used for debugging purposes only")
    log.info("This is a informational message")
    log.warn("This is a warning")
    log.error("This is an error message")
}
```

### Logger name

Each logger in the MWS logging configuration has a name. In the case of plugins, it is comprised of the full class name, including the package, prepended by "plugins.". For example, a plugin class of "example.LoggingPlugin" will have access to a logger configured as `plugins.example.LoggingPlugin`.

### Logging configuration

The logging configuration is done through the MWS configuration file. For more information on configuring loggers, see [Configuring Moab Web Services on page 1395](#). A good configuration for developing plugin types may be to add "plugins" at the debug level. Be sure to set the log level threshold down for the desired appender.

```
log4j = {
  ...
  // Appender configuration
  ...
  debug "plugins"
}
```

### Related topics

- [Plugin developer's guide on page 1679](#)

## i18n messaging

Plugins, translators, and custom components all have access to [i18n](#) messages. Utilizing messages requires the two following steps:

1. Including a file (or multiple files) that ends in "messages.properties" in the plugin JAR file.
2. Using the message method on a plugin type, translator, or custom component.

### [Including messages in plugin JAR file](#)

Messages are defined using property files. These may be named anything as long as they end with "messages.properties" and must be placed at the root or top level of the plugin JAR file. If they are present, they will be loaded automatically. Multiple property files may be used within a single plugin JAR file.

Each property file consists of an arbitrary amount of lines that define a message property (also called a code) with letters, numbers, and periods, associated with a human-readable message that can span multiple lines, have quotes, or contain arguments. These are demonstrated in the following example.

```
first.message.code=This is the first message
second.message=This message can span multiple lines, \\
  and will not show the linebreaks when retrieved
message.with.arguments=This message has arguments: first - {0}, second - {1}, third -
{2}, etc.
message.with.quotes=This message uses single quotes around 'this phrase'.
```

It is recommended to namespace the messages by using the property definitions and multiple property files if necessary. For example, suppose a plugin JAR existed which actually contained two plugin types: Message1Plugin and Message2Plugin. The first suggestion is to namespace the messages for each plugin by the property definition, such as the following:

```
message1Plugin.first.message=This is a message for Message1Plugin
message2Plugin.first.message=This is a message for Message2Plugin
```

These messages could be stored in a file named "messages.properties" in the root of the plugin JAR file. If there are many messages contained for each plugin type, it may be necessary to split each plugin type's messages into a separate file, such as "message1-messages.properties" and "message2-messages.properties". Note that it is essential that each property file ends with "messages.properties" so that it is registered correctly.



It is important that no two message codes are identical within a single plugin JAR file, even if they are defined in separate property files. If this is done, a conflict will exist with the messages and behavior is undefined.

### Using the message method

Each plugin, translator, and custom component is injected with a method named `message`. This method takes a `Map` as its parameter, which can contain one or several of the following properties:

Parameter	Type	Description
<b>code</b>	String	The message property definition (everything before the equals sign in the property file for a single message), for example, <code>first.message.code</code> .
<b>args</b>	List<Object>	A list of arguments to insert into the message.
<b>default</b>	String	A default message to be used when the message code cannot be resolved.
<b>error</b>	org.springframework.context.MessageSourceResolvable	An object that represents a hierarchy of message codes. This is typically used to display errors.

The most utilized parameters are `code` and `args`, as these combined provide great flexibility in generating messages. If a message cannot be resolved, or in other words the message definition does not exist, the code will simply be returned as the resolved message. Below are several examples of messages resolved using the property files given above. While these are contained in the polling method, the message may be used anywhere within a plugin type.

```
package example
import com.adaptc.mws.plugins.AbstractPlugin

class MessagingPlugin extends AbstractPlugin {
  def poll() {
    assert message(code:"first.message.code")== "This is the first message"
    assert message(code:"message.with.arguments", args:[
      "1st", 2, true
   ])== "This message has arguments: first - 1st, second - 2, third - true, etc."
    assert message(code:"message.with.quotes")== "This message uses single quotes around
    'this phrase'."
    assert message(code:"invalid.message.code")== "invalid.message.code"
  }
}
```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Configuration

Plugin types can access two different kinds of configuration: an individual plugin's configuration, and the global MWS application configuration.

### Individual plugin configuration

The individual plugin configuration is separate for each instance of a plugin. This may be used to store current configuration information such as access information for linked resources. It should not be used to store cached information or non-configuration related data. The individual datastore should be used instead for these cases (for more information, see [Individual datastore on page 1694](#)).

It is accessed by using the `getConfig` method discussed in [Dynamic methods on page 1681](#).

```
public void poll() {
    def configFromMethod = getConfig()
    // OR an even simpler method...
    def configFromMethod = config
}
```

A common case is to retrieve the configuration in the `configure` method, verify that it matches predetermined criteria, and utilize it perform initial setup of the plugin (e.g. initialize libraries needed to communicate with external resources). For example, to verify that the configuration contains the keys "username" and "password," the following code may be used.

```
public void configure() throws InvalidPluginConfigurationException {
    def myConfig = config
    // This checks to make sure the key exists in the configuration Map and that the
    // value is not empty or null
    if (!myConfig.containsKey("username") || !myConfig.username)
        throw new InvalidPluginConfigurationException("The username configuration parameter
        must be provided")
    if (!myConfig.containsKey("password") || !myConfig.password)
        throw new InvalidPluginConfigurationException("The password configuration
        parameter must be provided")
}
```

### Access MWS configuration

The MWS application configuration can also be accessed in plugin types. This configuration is global for the entire application and can be modified by the administrator as shown in [Configuring Moab Web Services on page 1395](#).

It is accessed by using the `getAppConfig` method discussed in [Dynamic methods on page 1681](#). This is demonstrated below:

```
public void poll() {
    // Retrieve the current MWS_HOME location
    def mwsHome = appConfig.mws.home.location
    // OR an even simpler method...
    def mwsHome = getAppConfig().mws.home.location
}
```

Any of the properties shown in the [Configuration](#) reference may be accessed. Custom properties may also be registered and accessed:

```
mws-config.groovy
-----
plugins.custom.property = "This is my custom property"
```

```
CustomAppPropertyPlugin
-----
public void poll() {
    assert appConfig.plugins.custom.property=="This is my custom property"
}
```

## Related topics

- [Plugin developer's guide on page 1679](#)

## Configuration constraints

Plugin types can optionally define validation constraints for the polling interval and plugin configuration. These parameters are then checked against the defined constraints during the creation of a new plugin. If the validation fails, meaning the configuration provided does not pass the constraints defined by the plugin type, the plugin will fail to be created with error messages based on the parameters and constraints defined.

### Defining constraints

To define constraints for a plugin type and therefore for all plugins created using it, use the following syntax:

```
import com.adaptc.mws.plugins.*
class ConstrainedPlugin extends AbstractPlugin {
    static constraints = {
        // Set plugin's default polling interval
        pollInterval defaultValue:60
        // The "myParam" configuration parameter is automatically required and cannot be
blank
        myParam blank:false
        // The "myEnum" configuration parameter is not required and must set to one of the
values in the list
        myEnum required:false, inList:["val1", "val2", "val3"]
        // Insert additional constraints here...
    }
}
```

In the table below, all available constraints are shown, as well as the expected value type, an example, the default message code, and the message suffix. The message columns are described in greater detail in the [Messaging](#) section below.

Constraint	Default value	Type	Example value	Default message code	Message suffix	Description
<b>blank</b>	--	Boolean	true	default.blank.message	blank	If false, the parameter (if present) cannot be a blank string.
<b>creditCard</b>	--	Boolean	true	default.invalid.creditCard.message	creditCard.invalid	If <b>true</b> , uses <code>org.apache.commons.validator.CreditCardValidator</code> to determine if the parameter (if present) is a valid credit card number.
<b>defaultValue</b>	--	Object or Closure	60	--	--	If the parameter is not present, it will be set to this default value. Does not return any error messages. See <a href="#">Default value</a> below for more information.
<b>email</b>	--	Boolean	true	default.invalid.email.message	email.invalid	If <b>true</b> , the parameter (if present) must be a valid email address.
<b>inList</b>	--	List	["first", "second"]	default.notinlist.message	notinList	The parameter (if present) must be set to one of the values specified.
<b>matches</b>	--	String	"[a-z][A-Z]+"	default.doesnt.match.message	matches.invalid	The parameter (if present) must match the specified regular expression.

Constraint	Default value	Type	Example value	Default message code	Message suffix	Description
<b>max</b>	--	Integer	10	default.invalid.max.message	max.exceeded	The parameter (if present) must not be greater than the defined value.
<b>*max-Size</b>	--	Integer	10	default.invalid.max.size.message	maxSize.exceeded	The parameter's (if present) size must not be greater than the defined value.
<b>min</b>	--	Integer	1	default.invalid.min.message	min.notmet	The parameter (if present) must not be less than the defined value.
<b>*min-Size</b>	--	Integer	1	default.invalid.min.size.message	minSize.notmet	The parameter's (if present) size must not be less than the defined value.
<b>notEqual</b>	--	Object	"Invalid Value"	default.not.equal.message	notEqual	The parameter (if present) must <i>not</i> be set to the defined value.
<b>nullable</b>	true	Boolean	false	default.null.message	nullable	If <b>true</b> , the parameter (if present) must be non-null value. See <a href="#">required</a> for how to enforce the parameter to be present.
<b>password</b>	--	Boolean	true	--	--	If <b>true</b> , the parameter (if present) is hidden from the user both on input and display when managing plugin configuration. It is not, however, hidden in the REST API. Does not return any error messages.

Constraint	Default value	Type	Example value	Default message code	Message suffix	Description
<b>range</b>	--	Range	1..10	default.invalid.range.message	range.toosmall/range.toobig	Uses a groovy range to validate that the value is within a specified range.
<b>required</b>	true	Boolean	false	default.required.message	required	If <b>true</b> , the parameter must be present and non-null for the plugin to be created successfully. Implies the <code>nullable:false</code> constraint.
<b>scale</b>	--	Integer	2	--	--	Only valid for Double parameters. Rounds the parameter (if present) to the specified number of digits. Does not return any error messages.
<b>*size</b>	--	Range	2	default.invalid.size.message	size.toosmall/size.toobig	Uses a groovy range to restrict the size of a collection, string, or a number.
<b>*type</b>	--	Classes	Integer.class	typeMismatch	typeMismatch	See <a href="#">Type inferencing and conversion</a> below.
<b>url</b>	--	Boolean	true	default.invalid.url.message	url.invalid	If <b>true</b> , uses <code>org.apache.commons.validator.UrlValidator</code> to determine if the parameter (if present) is a valid URL. Does not support <code>exec</code> or <code>file</code> scheme URLs.
<b>scriptable-Url</b>	--	Boolean	true	default.invalid.scriptable.url.message	scriptableUrl.invalid	Identical to the <a href="#">url</a> validator, but adds support for <code>exec</code> and <code>file</code> scheme URLs.

Constraint	Default value	Type	Example value	Default message code	Message suffix	Description
<b>validator</b>	--	Closure	(See <a href="#">Custom validator</a> )	default.invalid.validator.message	validator.error	See <a href="#">Custom validator</a> below.

\* The user interface (see [Plugin management on page 1742](#)) does not support parameters whose type is a subclass of Collection (a List, for example). Such parameters are therefore not recommended.

 The polling interval constraints must always apply to Integer types. If this specification is violated, the plugin type cannot be added or updated.

Messaging

When defined constraints are violated for a plugin, error messages are retrieved based on the configuration parameters and the applied constraints using i18n Messaging codes (see [i18n messaging on page 1683](#)). First, the most specific error message will be attempted to be resolved from a message code generated from the plugin type name, the configuration parameter, and the constraint. This code takes the format of `pluginTypeName.parameterName.suffix` where the plugin type's name has a lowercase first letter and the suffix is shown in the table above. If this message code is not defined, the default message code (as shown in the table above) will be used.

For example, if the [url](#) constraint validation failed for the "ExamplePlugin" plugin type's "endpoint" configuration parameter, the following message codes would be resolved in order:

- `examplePlugin.endpoint.url.invalid`
- `default.invalid.url.message`

 Plugin types that have two or more uppercase letters at the start of the name will not be converted to have a lowercase first letter for error message codes. In other words, for the example just given using "VCenterPlugin" instead of "ExamplePlugin", the following message codes would be resolved in order:

```
VCenterPlugin.endpoint.url.invalid
default.invalid.url.message
```

Default messages

Default messages may be contained in any `messages.properties` file included in the plugin JAR file as explained in i18n Messaging (see [i18n messaging on page 1683](#)). Arguments for each constraint vary, but they always include these argument indices:

- {0}: The configuration parameter name (for example, endpoint).
- {1}: The plugin type class name (for example, my.package.ExamplePlugin).
- {2}: The value of the configuration parameter.

If default messages are not defined in the plugin project, the following messages will be used:

```
default.doesnt.match.message=The '{0}' configuration parameter value ({2}) does not
match the required pattern '{3}'
default.invalid.url.message=The '{0}' configuration parameter value ({2}) is not a
valid URL
default.invalid.scriptable.url.message=The '{0}' configuration parameter value ({2})
is not a valid scriptable URL
default.invalid.creditCard.message=The '{0}' configuration parameter value ({2}) is
not a valid credit card number
default.invalid.email.message=The '{0}' configuration parameter value ({2}) is not a
valid e-mail address
default.invalid.range.message=The '{0}' configuration parameter value ({2}) does not
fall within the valid range from {3} to {4}
default.invalid.size.message=The '{0}' configuration parameter value ({2}) does not
fall within the valid size range from {3} to {4}
default.invalid.max.message=The '{0}' configuration parameter value ({2}) is greater
than the maximum value of {3}
default.invalid.min.message=The '{0}' configuration parameter value ({2}) is less
than the minimum value of {3}
default.invalid.max.size.message=The '{0}' configuration parameter value ({2})
exceeds the maximum size of {3}
default.invalid.min.size.message=The '{0}' configuration parameter value ({2}) is
less than the minimum size of {3}
default.invalid.validator.message=The '{0}' configuration parameter value ({2}) does
not pass custom validation
default.not.inlist.message=The '{0}' configuration parameter value ({2}) is not
contained within the list [{3}]
default.blank.message=The '{0}' configuration parameter cannot be blank
default.not.equal.message=The '{0}' configuration parameter value ({2}) cannot be
equal to '{3}'
default.null.message=The '{0}' configuration parameter cannot be null
default.required.message=The '{0}' configuration parameter is required and cannot be
null
typeMismatch=The '{0}' configuration parameter value ({2}) does not match the
required type '{3}'
```

### Labels and help messages

Message codes may also be provided for configuration parameters to aid the admin user with human readable property labels and help messages. Similar to the validation error message codes, labels and help message codes may be defined using the `pluginTypeName.parameterName.label` and `pluginTypeName.parameterName.help` message codes. These values are used only in plugin type management (see [Plugin type management on page 1737](#)) and are not exposed through the REST API.

### Type inferencing and conversion

Due to the dynamic nature of configuration parameters, the expected type or class of values for each parameter are inferred from constraints. The following rules govern how type is inferred, in priority order:

- If the **\*type** constraint is applied to a parameter, the constraint value will be used as the expected type.

**i** Only the `String`, `Date`, `Double`, `Integer`, and `Boolean` classes are supported for the **\*type** constraint. If `Float` or `Long` is desired, use `Double` and `Integer` respectively as the type.

- If the **inList** or **range** constraints are applied to a parameter, the class of the first element in the constraint value array is used as the expected type.
- If the **\*minSize** or **\*maxSize** constraints are applied to a parameter, `java.lang.Collection` is used as the expected type.
- If the **max**, **min**, or **notEqual** constraints are applied to a parameter, the class of the constraint value is used as the expected type.
- If none of the above apply, `java.lang.String` is used as the expected type.

If the configuration parameter values can be converted to the expected types, this will occur automatically. Otherwise, the **\*type** constraint is violated and the applicable error messages will be generated.

#### Custom validator

In cases where the built-in constraints prove inadequate for validation, custom validators may be used. The **validator** constraint expects a Groovy Closure parameter which has one or (optionally) two arguments: the value of the configuration parameter and the plugin object. With these parameters, complex validation logic may be defined. Additionally, custom message codes and arguments may be defined by validator constraints and these will be used in generating error messages when validation fails.

For example, suppose that the parameter "user" cannot be set to the same value as parameter "creator." Additionally, the "creator" parameter must not be equal to either "bob" or "joe." The existing constraints are inadequate to fulfill this use case, but the following code using validators would perform exactly as expected:

```
import com.adaptc.mws.plugins.*
class ConstrainedPlugin extends AbstractPlugin {
    static constraints = {
        user validator:{ val, obj ->
            if (val==obj.config.creator)
                return "invalid.equal.to.creator"
        }
        creator validator:{ val ->
            if ("val"=="joe")
                return ["invalid.equal", "joe"]
            if (val=="bob")
                return ["invalid.equal", "bob"]
        }
    }
}
```

In the examples above, the message codes and output on validation failure is shown below:

```
Message codes
-----
```

```
constrainedPlugin.user.invalid.equal.to.creator=The user configuration parameter value
({2}) must not be equal to the creator parameter.
constrainedPlugin.creator.invalid.equal=The creator configuration parameter must not
be equal to {3}.
```

```
Output error messages
-----
```

```
For user = "jill", creator = "jill"
"The user configuration parameter value (jill) must not be equal to the creator
parameter."
For user = "jill", creator = "bob"
"The creator configuration parameter must not be equal to bob."

For user = "jill", creator = "joe"
"The creator configuration parameter must not be equal to joe."
```

The validator Closure may return:

- `Nothing` (`null`) or `true` if the validation succeeded without errors.
- `false` if a validation error occurred (in this case the default validator message suffix would be used).
- A string which will be used as the message code suffix in the `pluginTypeName.propertyName.suffix` format.
- A list with the first element being the message code suffix, and all other elements being arguments for the message indexed starting at 3 (as shown in the example above).

All validator constraints automatically have the `appConfig` property available, which contains the application configuration as discussed in the Configuration section (see [Configuration on page 1685](#)). The `suite` property contains the value of the configured MWS suite. Additionally, services may be retrieved as explained in the next section.

### Retrieving services

At times it may be necessary to use Bundled Services in custom validators. A method named `getService` which takes a single string parameter of the name of the service (as used during injection) is provided to be used in these cases. For example, if a plugin needs a valid server certificate file, the SSL Service may be used as follows:

```
import com.adaptc.mws.plugins.*
class ConstrainedPlugin extends AbstractPlugin {
  static constraints = {
    certificateFile validator:{ val ->
      ISSLService sslService = getService("sslService")
      try {
        sslService.getSocketFactory(val)
      } catch(Exception e) {
        // Certificate file is invalid, return an error
        return ["invalid", e.message]
      }
    }
  }
}
```

**i** The `getService` method does not work with [translators](#), [custom components](#), [RM services](#), or the [Individual datastore](#) on page 1694.

### Default value

The default value for a configuration parameter might depend on the MWS configuration or other properties. Therefore, the [defaultValue](#) constraint can be set to a closure. The `defaultValue` closure does not take any parameters and must return the object to be used as the default value.

For example, if the default value of a parameter must be true if and only if MWS is configured for the Cloud suite, then the following constraints would satisfy these conditions:

```
import com.adaptc.mws.plugins.*
class ConstrainedPlugin extends AbstractPlugin {
  static constraints = {
    myParameter required: true, type: Boolean, defaultValue: {
      return suite == Suite.CLOUD
    }
  }
}
```

As with [validator](#) closures, `defaultValue` closures have access to `appConfig`, `suite`, and `getService`.

### Related topics

- [Plugin developer's guide on page 1679](#)

## Individual datastore

Each plugin has access to an individual, persistent datastore which may be used for a variety of reasons. The datastore is not designed to store Moab data such as nodes, jobs, or virtual machines, but custom, arbitrary data pertinent only to the individual plugin. This may include storing objects in a persistent cache, state information for currently running processes, or any other arbitrary data. The individual datastore has the following properties:

- Data is persisted to the Mongo database and will be available even if the plugin or MWS is restarted.

- The data must be stored in groups of data called **collections**. These correspond directly to MongoDB collections.
- Each plugin may have an arbitrary number of collections.
- Collections are guaranteed not to collide if there are identically named collections between two plugin types or even two plugin instances.
- Each collection contains multiple objects or **entries**. These correspond directly to MongoDB documents.
- The values of entries may be any object which can be serialized to MongoDB: simple types (int or Integer), Maps, and Lists.
- A collection is automatically created whenever an entry is added to it, it does not need to be specifically initialized.

To utilize the datastore, the [Plugin datastore service](#) must be used. Operations are provided to add, query, and remove data from each collection.

**i** Simple key/value storage is not currently provided with the datastore. It may easily be done, however, by storing data in the format of `{name:"key", value:"value"}` and then retrieving this entry later by querying on name equals "key."

### Example

The example below demonstrates two web services (see [Exposing web services on page 1696](#)). The first adds multiple entries containing various types of data to an arbitrarily named collection. The second retrieves the data and returns it to the user.

```
package example
import com.adaptc.mws.plugins.*

class DatastorePlugin extends AbstractPlugin {
  IPluginDatastoreService pluginDatastoreService

  def storeData(Map params) {
    def collectionName = params.collectionName
    def data = [[boolVal:true], [stringVal:"String"], [intVal:1], [nullVal:null]]
    if (pluginDatastoreService.addData(collectionName, data))
      log.info("Data successfully added")
    else
      log.info("There was an error adding the data")
    return [success:true]
  }

  def retrieveData(Map params) {
    def collectionName = params.collectionName
    return pluginDatastoreService.getCollection(collectionName)
  }
}
```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Exposing web services

Any number of methods may be exposed as public, custom web services by satisfying several criteria:

- The method must declare that it returns `Object` or `def`.
- The method must define a single argument of type `Map`.
- The method must actually return a `List` or `Map`.
- The method must not be declared as private or protected; only public or unscoped methods will be recognized as web services.

### Parameters and request body

The `Map` argument will contain all parameters passed into the web service by the client. See [Accessing Plugin Web Services on page 1605](#) for additional details.

Parameters may be passed into the web service call as normal URL parameters such as `?param=value&param2=value2`, as key-value pairs in the POST body of a request, or as JSON in the body.

For the first two cases, the parameters will be available on the `Map` argument passed into the web service call as key value pairs matching those of the request. Note that in these cases all keys and values will be interpreted as strings. However, the parameters object has several helper methods to convert from Strings to simple types, such as Booleans, integers, doubles, floats, and lists. If the value is not a valid simple type, null is returned.

Finally, note that the client may optionally include an `objectId` as the last part of the URL. When this is done, the `id` field will be set to this value in the `Map` argument to the web service.

```
GET <webServiceUrl>?key=value&key2=true&key3=5&list=1&list=2

def serviceMethod(Map params) {
    assert params.key=="value"
    assert params.key2=="true"
    assert params.bool('key2')==true
    assert params.key3=="5"
    assert params.int('key3')==5
    assert params.list('list')==[1, 2]

    // Null is returned if the conversion is invalid
    assert params.int('key')==null
}
```

When the body possesses JSON, the parsed JSON object or array will be available within a parameter called **body** in the `Map` argument. In this scenario, the types of the values are preserved by the JSON format.

```
POST <webServiceUrl> with JSON body of
{"key":"value","key2":true,"key3":5}

def serviceMethod(Map params) {
    assert params.body.key=="value"
    assert params.body.key2==true
    assert params.body.key3==5
}
```

### Unsecured web services

There are times when it is desirable to create a plugin with a publicly available web service that does not require a valid application account in order to access it (for details, see [Access control on page 1420](#)). In these cases, the [Unsecured](#) annotation may be used on the plugin web service method. No authentication will be performed on Unsecured web services. An example of using the annotation is given below.

```
Sample unsecured custom web service
-----
@Unsecured
def retrievePublicData(Map params) {
    return [data:["data item 1", "data item 2"]]
}
```



Be cautious in using this annotation as it may potentially present a security risk if sensitive data is returned from the web service.

### Returning errors

In order to signify an error occurred or invalid data was provided, the [WebServiceException](#) class may be thrown from any custom web service. This exception contains constructors and fields for a list of messages and a HTTP response code. For example, suppose that the user provided inadequate information. The web service could use the following code to notify the user and prompt them to take action with custom messages.

```
def service(Map params) {
    // Handle invalid input
    if (!params.int('a'))
        throw new WebServiceException("Invalid parameter 'a' specified, please specify an
integer!", 400)
    // Use params.a correctly ...
}
```

For the example above, a 400 response code (bad request) would be returned with a response body as follows:

```
{
  "messages":[
    "Invalid parameter 'a' specified, please specify an integer!"
  ]
}
```

If any other exception is thrown from a web service (ie Exception, IllegalArgumentException, etc.), a 500 response code will be returned with the following response body:

```
{
  "messages":[
    "A problem occurred while processing the request",
    "Message provided in the exception constructor"
  ]
}
```

See [Responses and return codes on page 1429](#) for more information on error formats in MWS.

### Accessing the HTTP Request Method

The HTTP method used for the request is available from the Map parameters argument. The key used to access it is stored as a static field in [PluginConstants](#) called `WEB_SERVICES_METHOD`. The value is a string which can be GET, POST, PUT, or DELETE. The following example demonstrates how this could be used with the `WebServiceException` to create a REST API with a plugin.

```
def serviceMethod(Map params) {
    // Check to make sure that this request used the HTTP GET method
    // Throw a 405 error (method not supported) if not
    if (params[PluginConstants.WEB_SERVICES_METHOD]!="GET")
        throw new WebServiceException("Method is not supported", 405)
}
```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Reporting state data

As long as Moab Workload Manager is configured with MWS as a Resource Manager (RM), plugins may report state information on jobs, nodes, storage, and virtual machines to Moab. This is done through **Reports** that are generated by the plugin and passed to the bundled RM services ([Job RM service](#), [Node RM service](#), [Storage RM service](#), and [Virtual machine RM service](#)). Each report is for a specific type of object: job, node, storage, or virtual machine. Each contains current state information on the specific attributes of the type it is for.

 Note that storage is a sub-type of node, meaning that it is a specialized node.

### Generating reports

To generate a report, simply create a new instance of a report depending on the type of object to be reported:

Object type	Report type
<b>Job</b>	<a href="#">JobReport</a>
<b>Node</b>	<a href="#">NodeReport</a>
<b>Storage</b>	<a href="#">StorageReport</a>
<b>Virtual Machine</b>	<a href="#">VirtualMachineReport</a>

Each report has a single required parameter for creating a new instance—the ID of the object which is being reported. Once the report instance has been created, any property may be modified as shown in the API documentation links in the table above. The following example shows the creation of a simple node report and modification of a few properties:

```

public void poll() {
    NodeReport node = new NodeReport("node1")
    node.timestamp = new Date()
    node.image = "centos-5.4-stateless"
    ... // Set other properties and persist the report
}

```

### Master and slave reports

At times, you may want to report some additional attributes on objects *only if* the objects are being reported by other plugins. For example, you may want to report the power state of a VM, but sometimes the plugin reporting this data can receive data even after the VM has been destroyed. In this case, you can set the `slaveReport` field on any report to `true`, signifying that the report should only be used if another plugin is reporting on the same object (in other words, creating "master" reports).

**i** If all reports for an object are "slave" reports, and no "master" reports exist, then the object will not report to Moab Workload Manager.

### Special cases in field values

All complex types, such as Lists, Maps, and objects (not including Enumerated values such as [NodeReportState](#) and [JobReportState](#)) have default values set for them and are not required to be instantiated before use. For example, the metrics property of a node report may be modified as follows:

```

public void poll() {
    NodeReport node = new NodeReport("node1")
    // The following assignments are equivalent in their functionality
    node.features.add("FEAT1")
    node.features << "FEAT2"
    // The following assignments are equivalent in their functionality
    node.metrics.METRIC1 = 4d
    node.metrics["METRIC2"] = 125.5
    ... // Set other properties and persist the report
}

```

For the resources and requirements (jobs only) properties, assignments may be made easily without checking for previously existing values or null objects. For example, resources may be added to the resources property simply by accessing it as a Map:

```

public void poll() {
    NodeReport node = new NodeReport("node1")
    node.resources.RES1.total = 10
    node.resources.RES1.available = 3
    node.resources["RES2"].total = 10
    node.resources["RES2"].available = 10
    ... // Set other properties and persist the report
}

```

The job report's requirements property has some additional handling to allow it to be accessed as a single [JobReportRequirement](#) object, such as in the following example:

```
public void poll() {
    JobReport job = new JobReport("job.1")
    job.nodeCountMinimum = 4
    job.processorCountMinimum = 2
    job.requiredNodeFeatures << "FEAT1"
    job.preferredNodeFeatures << "FEAT2"
    ... // Set other properties and persist the report
}
```

**i** Although multiple requirements may be added to the `requirements` list to provide consistency with the MWS Job resource (see [Jobs on page 1547](#)), only the first requirement object's properties will be reported to Moab through the RM interface.

Managing images for nodes

In order to have Moab Workload Manager recognize a node as a virtual machine hypervisor, it must have a valid associated Image (see [Images on page 1536](#)). In particular, the `image` property on a node report must set to a valid image name. The image's `hypervisorType` and `virtualizedImages` properties are then used to report the correct hypervisor type and supported virtual machine images to Moab.

If the `image` is invalid, it will be ignored and the node will not be recognized as a hypervisor. If the image is valid, but no `hypervisorType` value is present, the `extensions.xcat.hvType` field value will be used. If that is also not present, the configuration parameter for default hypervisor type (see [Configuration on page 1772](#)) will be used instead.

Persisting a Report

After a report has been generated and all desired fields have been updated, the report must be sent to one of the three bundled RM services for persisting. If this is not done, the report will be discarded and will not be considered when reporting state information to Moab. The RM services are shown below according to the object type that they handle:

Object type	RM service
<b>Job</b>	<a href="#">Job RM service</a>
<b>Node</b>	<a href="#">Node RM service</a>
<b>Storage</b>	<a href="#">Storage RM service</a>
<b>Virtual Machine</b>	<a href="#">Virtual machine RM service</a>

Each service has two methods: `save` and `update`. The difference between these is that the `save` method first removes all previous reports from the plugin calling the method, and then persists the new reports, thereby only persisting the latest reports, while the `update` method does not remove any reports before persisting the new reports. Typically, the `save` method will be used while a plugin is being polled, while the `update` method will be used in incremental event based reporting. An example of using the `save` method is shown below.

```
INodeRMService nodeRMService

public void poll() {
    NodeReport node = new NodeReport("node1")
    // Change the state
    node.state = NodeReportState.BUSY
    // Persist
    nodeRMService.save([node])
}
```

Once this is done, the reports will be persisted to MongoDB and will be included in RM queries (see [Resource manager queries on page 1735](#)) from Moab Workload Manager or users.

#### Related topics

- [Plugin developer's guide on page 1679](#)

## Controlling lifecycle

 Plugin control is currently in Beta. Interfaces may change significantly in future releases.

At times a plugin developer may wish to modify the current state of a plugin or even create plugins programmatically. This may be done with the [Plugin control service](#). Operations exist on the service to:

- create plugin instances dynamically with specific configuration.
- retrieve plugin instances by ID or based on configuration properties.
- start or stop plugin instances.
- verify plugin instance configuration.

#### Creating plugins

Several methods are provided to allow on-the-fly creation of new plugins. Generally, they allow a plugin with a specific ID and plugin type (as a string or as a Groovy Class) to be created with optional configuration properties. These properties should match the fields in [Plugins on page 1599](#).

If any configuration properties are omitted, the defaults will be used as described in [Setting default plugin configuration on page 1747](#). A boolean value is also returned indicating whether the creation succeeded or not.

Note that the `createPlugin` methods will initialize the plugin for retrieval or usage and attempt to start the plugin if the `autoStart` property is true.

#### Retrieving plugins

Plugins may be retrieved by using an ID, querying by plugin type, or even querying based on configuration parameters. Several methods are provided to perform these functions as shown on [Plugin control service on page 1750](#).

#### Starting and stopping plugins

Plugins may also be started or stopped on demand. These two methods are exposed directly as `start` and `stop` on the plugin control service. Although each method does not return any data, exceptions are

thrown if errors are encountered.

### Verifying plugin configuration

Finally, the plugin control service may be used to verify plugin configuration at any point instead of just when the plugin is started or modified. This may be useful to attempt to modify plugin configuration directly through the `setConfig` dynamic method (see [Dynamic methods on page 1681](#)) and then verify that the new configuration is valid for the plugin. Exceptions are thrown if the plugin or the configuration is invalid.

### Examples

If an error state is detected it may be necessary to stop the current plugin instance until corrective action can be taken. This may be done using the following code:

```
package example

import com.adaptc.mws.plugins.*

class ErrorPlugin {
    IPluginControlService pluginControlService

    public void poll() {
        // Error is detected, stop plugin instance!
        try {
            log.warn("An error was detected, trying to stop the plugin ${id}")
            pluginControlService.stop(id)
            log.warn("The plugin was successfully stopped")
        } catch (PluginStopException e) {
            log.error("Plugin instance ${id} could not be stopped", e)
        }
    }
}
```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Accessing MWS REST resources

Often a plugin type may need to access existing MWS REST Resources in order to extend or complement default MWS functionality. This may be done with the [Moab REST service](#), which allows a plugin type developer to utilize the existing Resources documentation see [Resources introduction on page 1446](#)) to perform these tasks.

All accesses to resources require a HTTP method to use (such as GET, POST, PUT, or DELETE) and a relative URL (such as `/rest/jobs`). Although it mimics the REST resource interface, no actual requests are made and no data is transmitted through the network.

### Authentication

All resources are available to the Moab REST Service, and no authentication or Application Accounts are needed.

**i** Caution must be used when developing plugin types, as there are no restrictions to what may be done with the Moab REST Service. This is especially true when not utilizing hooks as discussed below.

### Hooks

If pre and post-processing hooks are utilized in MWS ([Pre and post-processing hooks on page 1434](#)), the plugin type developer may choose whether or not they are executed when performing a "request" through the Moab REST service. This is done through the hooks option as documented in [Moab REST service on page 1748](#).

### Verifying API version support

The Moab REST Service provides a method for easily determining which API versions are supported by the current version of MWS. This method includes checks to make sure that the API version will work as expected, including verifying any configuration or external services are running.

```
moabRestService.isAPIVersionSupported(1)
moabRestService.isAPIVersionSupported(2)
```

### Converting string dates

Because the Moab REST Service returns data exactly as given to an external consumer of MWS, including dates converted to strings, the service provides a method for converting MWS date strings to actual Date objects.

```
moabRestService.convertDateString("2011-11-08 13:18:47 MST")
```

### URL parameters

URL parameters, such as `query`, `sort`, `proxy-user`, and others should be not be appended directly to the URL. Instead, these may be specified with the `params` option:

```
// Query images that are hypervisors
moabRestService.get("/rest/images", params:[query: '{"hypervisor":true}'])
// Sort images by osType
moabRestService.get("/rest/images", params:[sort: '{"osType":1}'])
```

### Examples

This code retrieves a list of all nodes, and is equivalent to the [Get all nodes](#) task.

```

package example

import com.adaptc.mws.plugins.*
import net.sf.json.*

class RestPlugin {
    IMoabRestService moabRestService

    public void poll() {
        def result = moabRestService.get("/rest/nodes")
        // OR with the hook enabled..
        def result = moabRestService.get("/rest/nodes", hooks:true)

        assert result instanceof MoabRestResponse
        assert nodes instanceof List

        log.debug("Nodes list:")
        nodes.each { JSON node ->
            log.debug(node.id)
        }
    }
}

```

This code adds a flag to a job, and is equivalent to the [Modify job attributes](#) task. This request also enables the hook (if one is configured) for the "request" and uses a URL parameter. This is the equivalent of making a call to `/rest/jobs/job.1?proxy-user=adaptive`.

```

package example

import com.adaptc.mws.plugins.*
import net.sf.json.*

class RestPlugin {
    IMoabRestService moabRestService

    public void poll() {
        def jobId = "job.1"
        def result = moabRestService.put("/rest/jobs/"+jobId, hooks:true, params:['proxy-
user':'adaptive']) {
            [flags:["RESTARTABLE"]]
        }
        assert result.isSuccess()
    }
}

```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Creating events and notifications

Plugins may easily create new events and create or update notification conditions using the [Plugin event service](#). Previously, this was only possible by utilizing the MWS REST resources. The event service eases this burden from plugin developers. There are several operations that are available using the service:

- Create an event with or without specifying an event date.
- Create an event from an enumeration annotated with `EventEnumeration` (see [Plugin event service on page 1755](#)) with or without specifying an event date.
- Create or update a notification condition with or without specifying an observed date or expiration duration.

## Creating events

Events are composed of several properties such as arguments, associated objects, origin, message, severity, escalation level, and a unique event code. The plugin event service removes the need for magic strings such as those for event severity ("INFO", "WARN", "FATAL") and also handles creating unique event codes. In other words, no bitwise manipulation is required to create new events.

The event code is comprised of several elements:

Code element	Description
Severity	If the event is informational, a warning, an error, or fatal.
Escalation level	Who cares about the event, or who should act on the event.
Component code	Internally made up of the MWS component code (stored internally) and the plugin event component code (see <a href="#">Plugin event component code on page 1705</a> ).
Entry code	The code representing a unique event for the component (for each plugin event component code).

The plugin event service handles the severity, escalation level, and entry code portions of the code by the values passed as parameters to the `createEvent` method. The plugin event component code is described in the next section.

### [Plugin event component code](#)

The plugin event component code should be a unique number across all plugin types or projects from 1–254. This number is combined with the MWS component code to represent each plugin as a unique component code across all Adaptive Computing products. 0 is reserved for MWS itself and should not be used. 255 is reserved for plugin types that do not define an event component code and represents an "unknown" plugin component. Additionally, codes 1–150 are reserved for Adaptive Computing plugins, while 151–254 are reserved for Professional Services and/or customer-specific plugins.

This code may be specified by setting an `eventComponent` property (see [Fields: Plugin Types on page 2204](#)) on the plugin project file or as a static property on the plugin type. As with all other project properties, the plugin type value overrides the project value. For example:

```

class MyExampleProject {
    ...
    Integer eventComponent = 2
    ...
}
ExamplePlugin {
    static final eventComponent = 1
    ...
}
Example2Plugin {
    // no eventComponent property
    ...
}

```

In this case, the plugin type `ExamplePlugin` has a plugin event component code of 1, while the `Example2Plugin` has a code of 2 since it inherits it from the project properties.

### Origin suffix

The origin of an event created through the plugin event service is automatically set by the plugin framework to `MWS/plugins/<plugin type>/<plugin id>`. For example, an event created by the plugin created from the "ExamplePlugin" plugin type with an ID of "plugin1" would generate events with an origin of `MWS/plugins/Example/plugin1`.

While this origin is sufficient for an administrator to determine the plugin where the event came from, the plugin developer may want this to be more specific to a class name or method name. This may be done using the optional `originSuffix` parameter to the `createEvent` method. The origin suffix, as its name implies, is appended to the end of the generated origin. For the example above, suppose the plugin developer passed `myMethod/switch1` as the origin suffix parameter when creating a new event. The event would then have an origin of `MWS/plugins/Example/plugin1/myMethod/switch1`.

### Event enumerations

While creating events using the plugin event service is quite simple, often there are related events that have properties in common, such as the event type prefix or the origin suffix. Additionally, [i18n messages](#) (see [i18n messaging on page 1683](#)) are typically used for the event's message. Using the `EventEnumeration` annotation (see [Plugin event service on page 1755](#)) in combination with an enumeration simplifies this process. When this is done, each message is pulled from the `messages.properties` files using a standard convention, and the event type prefix and the origin suffix may optionally added as static properties on the enumeration. Using `EventEnumeration` requires:

- The annotated element is an `enum`, not a class or interface.
- Each enumeration value must use the constructor with three arguments: the event name, the severity, and the escalation level.
- If an event type prefix is specified, it must be defined as `"static String EVENT_TYPE_PREFIX = ..."`, otherwise the property should not be defined.
- If an origin suffix is specified, it must be defined as `"static String ORIGIN_SUFFIX = ..."`, otherwise the property should not be defined.

If any of these conditions are not fulfilled, using the `EventEnumeration` annotation will result in compilation errors.

Enumeration values are automatically marked as implementing the `IPluginEvent` interface and may be used as the first parameter of the `createEvent` method on the plugin event service. For example:

```
package example

import com.adaptc.mws.plugins.EventEnumeration
import com.adaptc.mws.plugins.IPluginEventService.AssociatedObject
import static com.adaptc.mws.plugins.IPluginEventService.Severity.*
import static com.adaptc.mws.plugins.IPluginEventService.EscalationLevel.*

public class ExamplePlugin {
    void poll() {
        // Event 1 takes no arguments
        pluginEventService.createEvent(ExampleEvents.EVENT1, null, null)
        // Event 2 takes one argument and has an associated object
        pluginEventService.createEvent(ExampleEvents.EVENT2, ["arg1"], [new AssociatedObject
(type:"type1", id:"id1")])
    }
}

@EventEnumeration
enum ExampleEvents {
    EVENT1("Example One", INFO, USER), // Entry code is 0
    EVENT2("Example Two", INFO, USER) // Entry code is 1
}
```

It may be noted that several key properties of events are missing from the enumeration definition and create event call parameters:

- Message: retrieved automatically from `i18n` messages (see [Messages for event enumerations on page 1707](#))
- Event type: generated from the enumeration constructor and optional event type prefix property (see [Event type for event enumerations on page 1708](#))
- Entry code: generated from the return value of `ordinal()` on the enumeration value; in other words, this is generated from the order of the enumeration values

### Messages for event enumerations

The message for events created from enumerations is generated using `i18n` messages (see [i18n messaging on page 1683](#)) with codes in the following format:

- `<enumeration type name>.<enumeration value name>.message`
- `<enumeration type name>.<enumeration value name>.comment`

Considering the example in the section above, the message for `ExampleEvents.EVENT1` would be generated using the argument list passed to the `createEvent` method with the `"ExampleEvents.EVENT1.message"` message from `messages.properties`. This message should contain arguments if needed, such as `"My example with ID {0} was created"` and is used as the `"message"` property in the created event. The comment, on the other hand, is not persisted with the event and should be text (typically in paragraph format) describing why the event typically occurs or what actions should be taken when it does occur. Consider the message to contain instance specific information for the event (passed as arguments to the message) and the comment to be general documentation concerning the event.

As a best practice, name event enumeration values using the number and short name of each argument to the message. This makes it easy for the consumer to know which arguments are expected and what each means. For example, if an event is for connection errors and needs two arguments to the message, the URL and the error message, the enumeration value should be named "CONNECT\_FAILURE\_1URL\_2ERROR" or even "CONNECT\_TO\_1URL\_FAILURE\_2ERROR". In this way, the consumer knows that the first argument represents the URL and the second is the error message.

### Event type for event enumerations

As described above, the static string field `EVENT_TYPE_PREFIX` may be defined on the enumeration. This value is optional and, when present, is prepended with a space to the event name parameter from the constructor to generate the event type. For example, consider the following enumeration:

```
package example

import com.adaptc.mws.plugins.EventEnumeration
import static com.adaptc.mws.plugins.IPluginEventService.Severity.*
import static com.adaptc.mws.plugins.IPluginEventService.EscalationLevel.*

@EventEnumeration
enum MyPluginEvents {
    CONNECT("Connect", INFO, ADMIN),
    DISCONNECT("Disconnect", INFO, ADMIN)

    static String EVENT_TYPE_PREFIX = "My Plugin"
}
```

If `MyPluginEvents.CONNECT` and `MyPluginEvents.DISCONNECT` were used with the plugin event service, the generated event types would be "My Plugin Connect" and "My Plugin Disconnect" respectively.

### Origin for event enumerations

The origin for event enumeration values automatically contains more information than those for non-enumerated events, such as those described above. The enumeration type name and value are appended to the origin. For example, consider the following enumeration and plugin fragment:

```
...
class ExamplePlugin {
    ...
    assert id=="example1" // plugin ID is example1
    pluginEventService.createEvent(ExampleEvents.EVENT1, null, null)
    ...
}
...
@EventEnumeration
enum ExampleEvents {
    EVENT1("Event One", INFO, ADMIN)
    ...
}
```

The origin generated for the created event would be `MWS/plugins/Example/example1/ExampleEvents/EVENT1`. The static string field `ORIGIN_SUFFIX` may also be defined on the enumeration. This value is optional and, when present, is appended to the end of the generated origin as described above with the origin suffix parameter to the `createEvent` method.

Example

In order to understand all interactions when event enumerations are used, the following is a complete example.

## Plugin type

```

package example
import com.adaptc.mws.plugins.*

class ConnectPlugin extends AbstractPlugin {
    static eventComponent = 1

    IPluginEventService pluginEventService

    void poll() {
        def errorMessage = connect()
        if (errorMessage)
            pluginEventService.createEvent(ConnectEvents.CONNECT_TO_1URL_FAILURE_2ERROR,
[config.url, errorMessage], null)
        else
            pluginEventService.createEvent(ConnectEvents.CONNECT_SUCCESS, null, null)
    }

    // Returns the error message or null/empty on success
    private String connect() {
        String errorMessage
        ...
        return errorMessage
    }
}

```

## Event enumeration

```

package example
import com.adaptc.mws.plugins.EventEnumeration
import static com.adaptc.mws.plugins.IPluginEventService.Severity.*
import static com.adaptc.mws.plugins.IPluginEventService.EscalationLevel.*

@EventEnumeration
enum ConnectEvents {
    CONNECT_SUCCESS("Success", INFO, ADMIN),
    CONNECT_TO_1URL_FAILURE_2ERROR("Failure", ERROR, ADMIN)

    static String EVENT_TYPE_PREFIX = "Connect"
}

```

```

messages.properties
-----
ConnectEvents.CONNECT_SUCCESS.message=The plugin was successfully connected!
ConnectEvents.CONNECT_SUCCESS.comment=This occurs when the plugin successfully
connects to the configured URL and
    is informational only.
ConnectEvents.CONNECT_TO_1URL_FAILURE_2ERROR.message=The plugin failed to connect to
{0}: {1}
ConnectEvents.CONNECT_TO_1URL_FAILURE_2ERROR.comment=This occurs when the plugin fails
to connect to the configured
    URL for any reason. The most common reason is that the service is not running and
needs to be started.
    
```

The following are examples of the events created in MWS:

```

Created events
-----
{"totalCount": 2, "resultCount": 2, "results": [
  {
    "arguments": ["http://localhost:1000", "The service is not running!"],
    "code": 570523649,
    "eventDate": "2013-06-12 19:16:50 UTC",
    "eventType": "Connect Failure",
    "message": "The plugin failed to connect to http://localhost:1000: The service is
not running!",
    "origin": "MWS/plugins/Connect/connect/ConnectEvents/CONNECT_TO_1URL_FAILURE_
2ERROR",
    "severity": "ERROR",
    "id": "51b8c922a816c6a04af2401d",
    "associatedObjects": []
  },
  {
    "arguments": [],
    "code": 33652736,
    "eventDate": "2013-06-12 19:18:07 UTC",
    "eventType": "Connect Success",
    "message": "The plugin was successfully connected!",
    "origin": "MWS/plugins/Connect/connect/ConnectEvents/CONNECT_SUCCESS",
    "severity": "INFO",
    "id": "51b8c96fa816c6a04af24021",
    "associatedObjects": []
  }
]}
    
```

Unique event codes

The last topic that must be covered in creating events from plugins is that all efforts should be made to make sure that event codes are unique throughout all Adaptive Computing product suites. Additionally, the codes should be static, meaning they do not change once established. In order to do this, adhere the following recommendations:

- Use a unique (across all plugin types) plugin event component code for each plugin type.
- Follow the guidelines for plugin event component codes established above (see [Plugin event component code on page 1705](#)) and ensure it is a number 1-254.

- Use event enumerations where possible, otherwise ensure (through testing if possible) that all entry codes are unique for each plugin type.
- Ensure (through testing if possible) that the ordinal value of the event enumeration values do not change.

## Creating or updating notification conditions

The plugin event service also makes it easy to create or update notification conditions. Simply use the `updateNotificationCondition` method. Just as the MWS notification condition resource, this is an idempotent operation, meaning it can be called multiple times with the same result. If the notification condition does not exist, it will be created automatically. If it does exist, the observed date and details will be updated accordingly.

## Examples

Examples are available on [Plugin event service on page 1755](#).

### Related topics

- [Resources introduction on page 1446](#)
- [Events on page 1528](#)
- [Notifications on page 1585](#)
- [Notification conditions on page 1580](#)
- [Plugin developer's guide on page 1679](#)
- [Fields: Events on page 1894](#)
- [Plugin event service on page 1755](#)
- [Handling events on page 1711](#)
- [System events on page 1444](#)
- [Securing the connection with the message queue on page 1417](#)

## Handling events

**i** Plugin events (excepting the poll event) are currently in Beta. Interfaces may change significantly in future releases.

Plugin types may handle specific events by containing methods defined by the conventions below. All events are optional.

### The polling event

To maintain current information, each plugin is polled at a specified time interval. The following method definition is required to utilize the polling event.

```
void poll() { ... }
```

Typically this polling method is used to report node and virtual machine information. By default, the polling interval is set to 30 seconds, but can be modified for all or individual plugins as explained in [Plugin management on page 1742](#).

When a polling event occurs, the `poll` method on the target plugin is called. This method may perform any function desired and should typically make calls to the [Node RM service](#), the [Virtual machine RM service](#), and the [Job RM service](#) services to report the current state of nodes and virtual machines. For example, the `poll` method in the Native plugin type is implemented as follows:

**i** This is an extremely simplified version of what is actually implemented in the Native plugin type.

```
INodeRMService nodeRMService;
IVirtualMachineRMService virtualMachineRMService;

public void poll() {
    nodeRMService.save(getNodes());
    virtualMachineRMService.save(getVirtualMachines());
}
```

This simple poll method calls two other helper methods called `getNodes` and `getVirtualMachines` to retrieve node and virtual machine reports. These reports are then sent to the appropriate RM service. See [Reporting state data on page 1698](#) for more information on the RM services; however, the objective of this example is to demonstrate one possible use of the poll event handler. Other plugin types, on the other hand, may use the poll event to update internal data from pertinent resources or make calls to external APIs.

[Lifecycle events](#)

Events are also triggered for certain lifecycle state changes. The following method definitions are required to receive lifecycle events.

```
public void configure() throws InvalidPluginConfigurationException { ... }
public void beforeStart() { ... }
public void afterStart() { ... }
public void beforeStop() { ... }
public void afterStop() { ... }
```

Each event is described in the table below with the associated state change when the event is triggered.

State change	Event	Description
<b>configure</b>	Configure	Triggered before <code>beforeStart</code> and after the plugin has been configured. May be used to verify configuration and perform any setup needed any time configuration is loaded or modified.
<b>beforeStart</b>	Start	Triggered just before starting a plugin.
<b>afterStart</b>	Start	Triggered just after a plugin has been started.

State change	Event	Description
<b>beforeStop</b>	Stop	Triggered just before stopping a plugin.
<b>afterStop</b>	Stop	Triggered just after stopping a plugin.

Currently, no events are triggered for pausing, resuming, erroring, or clearing errors for plugins.

### RM events

When MWS is configured as a Moab Resource Manager (see [Moab Workload Manager resource manager integration on page 1730](#), and more specifically, [Configuring Moab Workload Manager on page 1731](#)), RM events are sent from Moab to each plugin according to the routing specification (see [Routing on page 1679](#)). The following method definitions are required to receive these events.

```
public boolean jobCancel(String jobName) { ... }
public boolean jobModify(String jobName, Map<String, Object> attributes, ModifyMode
modifyMode) { ... }
public boolean jobRequeue(String jobName) { ... }
public boolean jobResume(String jobName) { ... }
public boolean jobStart(String jobName, List<String> nodes, String username) { ... }
public boolean jobSubmit(Map<String, Object> job, String submissionString, String
submissionFlags) { ... }
public boolean jobSuspend(String jobName) { ... }
public boolean nodeModify(List<String> nodes, Map<String, String> attributes,
ModifyMode modifyMode) { ... }
public boolean nodePower(List<String> nodes, NodeReportPower state) { ... }
public boolean virtualMachinePower(List<String> virtualMachines, NodeReportPower
state) { ... }
```

### Related topics

- [Events on page 1528](#)
- [Notifications on page 1585](#)
- [Notification conditions on page 1580](#)
- [Plugin developer's guide on page 1679](#)
- [Fields: Events on page 1894](#)
- [Resources introduction on page 1446](#)
- [Plugin event service on page 1755](#)
- [Creating events and notifications on page 1704](#)

## Handling exceptions



Plugin exceptions are currently in Beta. Interfaces may change significantly in future releases.

The `com.adaptc.mws.plugins` package contains several exceptions that may be used and in some cases, should be caught. All exceptions end with "Exception", as in [PluginStartException](#).

There are several specific cases where Exceptions should or can be used:

- The `reload` method on the [Plugin control service](#) can throw the [InvalidPluginConfigurationException](#) to signify that the configuration contains errors.
- Various methods on the [Plugin control service](#) throw plugin exceptions which must be caught to diagnose errors when creating plugin types.
- Any exception (including the Exception class) can be thrown from a custom web service to display a 500 Internal Server Error to the client requesting the service with the given error message.

#### Related topics

- [Plugin developer's guide on page 1679](#)

## Managing SSL connections

At times it is desirable to load and use self-signed certificates, certificates generated from a single trusted certificate authority (CA), or even simple server certificates. It may also be necessary to use client certificates to communicate with external resources. To ease this process, the [SSL service](#) may be utilized. This service provides methods to load client and server certificates from the filesystem. Methods are also present to aid in creating connections which automatically trust all server certificates and connections.

Several points should be noted when using the SSL Service:

- Certificate files may be in the PEM file format and do not need to be in the DER format (as is typical of Java security).
- Each method returns an instance of `SSLSocketFactory`, which may then be used to create simple sockets or, in combination with another client library of choice, create a connection.
- If the client certificate password is non-null, it will be used to decrypt the protected client certificate.
- This service is *not* needed when performing SSL communications with trusted certificates, such as those for HTTPS enabled websites that do not have a self-signed certificate.
- If the file name of the certificate file (client or server) is relative (no leading '/' character), it will be loaded from the `mws.certificates.location` configuration parameter (see [Configuration on page 1772](#)).
  - The default value of `mws.certificates.location` is `MWS_HOME/etc/ssl.crt`.
- Both the client certificate alias and password may be `null`. In this case, the client certificate must not be encrypted and the client certificate's default alias (the first subject CN) will be used.
- The lenient socket factory and hostname verifier automatically trust all server certificates. Because of this, they present a large security hole. Only use these methods in development or in fully trusted environments.

#### Example

To create a socket to a server that requires a client certificate, the following code may be used.

```

package example

import com.adaptc.mws.plugins.*

class SSLConnectionPlugin extends AbstractPlugin {
  ISslService sslService

  public void poll() {
    // This certificate is not encrypted and will be the only certificate presented to
    the
    // connecting end of the socket.
    // This file will be loaded from MWS_HOME + mws.certificates.location + my-cert.pem.
    String clientCert = "my-cert.pem"

    def socketFactory = sslService.getSocketFactory(clientCert, null, null)
    def socket = socketFactory.createSocket("hostname.com", 443)
    // Write and read from the socket as desired..
  }
}

```

To create a HTTPS URL connection to a server that has a self-signed certificate, the following code may be used. Note that this is very typical of client libraries – they have a method to set the SSL socket factory used when creating connections.

```

package example

import com.adaptc.mws.plugins.*

class SSLConnectionPlugin extends AbstractPlugin {
  ISslService sslService

  public void poll() {
    // This certificate represents either the server public certificate or the CA's
    certificate.
    // Since the path is absolute it will not be loaded from the MWS_HOME directory.
    String serverCert = "/etc/ssl/certs/server-cert.pem"

    def socketFactory = sslService.getSocketFactory(serverCert)

    // Open connection to URL
    HttpURLConnection conn = "https://hostname.com:443/test".toURL().openConnection()
    conn.setSSLSocketFactory(socketFactory)

    // Retrieve page content and do with as desired..
    def pageContent = conn.getInputStream().text
  }
}

```

#### Related topics

- [Plugin developer's guide on page 1679](#)

## Utilizing services or custom "helper" classes

There are three general types of services available for use in plugins:

- Bundled services such as the [Moab REST service](#).
- Custom built translators loaded by convention of their name.

- Other custom built helper classes registered with Annotations.

These will each be described in this section.

## Bundled services

Bundled services are utility classes that are included and injected by default onto all plugin types. It is not required to use any of these services, but they enable several core features of plugin types as discussed in [Utility services on page 1677](#).

More information may be found on each bundled service in [Plugin services on page 1747](#).

## Using translators

Often a plugin type class file becomes so complex that it is desirable to split some of its logic into separate utility service classes. The most typical use case for this is to split out the logic for "translating" from a specific resource API to a format of data that the plugin type can natively understand and utilize. For this reason, there is a convention defined to easily add these helper classes called "Translators."

Simply end any class name with "Translator," and it will be automatically injected just as bundled services onto plugin types, other translators, or even custom registered components. The injection occurs only if a field exists on the class matching the name of the translator with the first letter lower-cased. For example, a translator class called "MyTranslator" would be injected on plugin types, other translators, and custom components that define a field called "myTranslator" as `def myTranslator` or `MyTranslator myTranslator`.

 Do not use two upper-case letters to start the class name of a Translator. Doing this may cause injection to work improperly. For example, use `RmTranslator` instead of `RMTranslator` as the class name.

 Be careful not to declare translator and custom component injection such that a cyclic dependency is created.

### [Logging in translators](#)

All translators automatically have a "getLog" method injected on them which can be used to access the configured logger. It returns an instance of [org.apache.commons.logging.Log](#).

```
package example

class ExampleTranslator {
    public void myMethod() {
        // log will be translated to getLog() by the groovy compiler
        log.info("Starting my method")
    }
}
```

See [Logging on page 1682](#) for more information on logging configuration and usage.

### Example

Suppose that a translator needs to be created to handle a connection to access an external REST resource. The translator could be defined as follows:

```
package example

class ExampleTranslator {
    public int getExternalNumber() {
        def number = ... // Make call to external resource
        return number
    }
}
```

A plugin type can then use the translator by defining a field called "exampleTranslator". Note that an instance does not need to be explicitly created.

```
package example

class ExamplePlugin {
    def exampleTranslator
    // OR ...
    //ExampleTranslator exampleTranslator

    public void poll() {
        // Use the translator
        log.info("The current number is "+exampleTranslator.getExternalNumber())
    }
}
```

To extend the example, the translator may also be injected into another translator:

```
package example

class AnotherTranslator {
    def exampleTranslator

    public int modifyNumber(int number) {
        return number + exampleTranslator.getExternalNumber()
    }
}
```

This translator may be used in the plugin type just as the other translator.

## Registering custom components

There are cases where the concept of a "Translator" does not fit the desired use of a utility class. In these cases, it is possible to register any arbitrary class as a component to be injected just as a translator would be. This is done using the Spring Framework's annotation `org.springframework.stereotype.Component`. When this annotation is used, the class is automatically registered to be injected just as translators onto plugin types and translators.



All annotations are available in the dependencies declared by the plugins-commons artifact.

**i** Do not use two upper-case letters to start the class name of a custom component. Doing this may cause injection to work improperly. For example, use `RmUtility` instead of `RMUtility` as the class name.

### Changing scope

By default, when a custom component is injected, only a single instance is created for all classes which inject it. This is referred to as the 'singleton' scope. Another scope that is available is 'prototype', which creates a new instance every time it is injected. This is useful when the class contains state data or fields that are modified by multiple methods. To change the scope, use the `org.springframework.context.annotation.Scope` on the class with a single `String` parameter specifying "singleton" or "prototype."

### Injecting translators or components

The need may arise to inject translators or other custom components onto custom components. This is done using the `org.springframework.beans.factory.annotation.Autowired` or `javax.annotation.Resource` annotations. The `Autowired` annotation is used to inject class instances by the type (i.e. `MyTranslator myTranslator`) while the `Resource` annotation is used to inject class instances by the name (i.e. `def myTranslator`). Add the desired annotation to the field that needs to be injected.

**i** Note that using the `Autowired` annotation does injection by type which differs from translator and plugin type injection. These are done by name just as the `Resource` annotation allows. Due to this fact, a type of "def" cannot be used when doing injection onto custom components using the `Autowired` annotation. See the example below.

Injection of custom components *onto* translators and plugin types are still done by name, only fields injected using the `Autowired` annotation are affected.

**!** Be careful not to declare translator and custom component injection such that a cyclic dependency is created.

### Logging in custom components

Unlike plugins and translators, custom components do *not* automatically have a "getLog" method injected on them. In order to log with custom components, you must use the Apache Commons Logging classes to retrieve a new log. The `PluginConstants` class contains the value of the logger prefix that is used for all plugins and translators. The following is an example of how to retrieve and use a logger correctly in a custom component.

```

package example

import com.adaptc.mws.plugins.PluginConstants
import org.apache.commons.logging.Log
import org.apache.commons.logging.LogFactory
import org.springframework.stereotype.Component

@Component
class ExampleComponent {
    private static final Log log = LogFactory.getLog(PluginConstants.LOGGER_
PREFIX+this.name)

    public void myMethod() {
        log.info("Starting my method")
    }
}

```

See [Logging on page 1682](#) for more information on logging configuration and usage.

### Example

Suppose that a custom utility class is needed to perform complex logic. A custom component could be defined as follows (notice the optional use of the Scope annotation):

```

package example

import org.springframework.stereotype.Component
import org.springframework.context.annotation.Scope

@Component
@Scope("prototype")
class ComplexLogicHandler {
    def handleLogic() {
        ... // Perform complex logic and return
    }
}

```

A plugin type or translator could then be defined to inject this component:

```

package example

class CustomPlugin {
    def complexLogicHandler

    public void poll() {
        complexLogicHandler.handleLogic()
    }
}

```

Now suppose another custom component needs to use the ComplexLogicHandler in its code. It can inject it using the Autowired annotation:

```

package example

import org.springframework.stereotype.Component
import org.springframework.beans.factory.annotation.Autowired

@Component
class AnotherHandler {
    // Note that this is injected by type, so 'def' may not be used
    @Autowired
    ComplexLogicHandler complexLogicHandler

    def wrapLogic() {
        complexLogicHandler.handleLogic()
    }
}

```

To perform the same injection but by name (as translators and plugin types are injected), use the `Resource` annotation:

```

package example

import org.springframework.stereotype.Component
import javax.annotation.Resource

@Component
class AnotherHandler {
    // Note that this is injected by name based solely on the name defined in
    // the annotation. The name of the field itself does not affect the injection.
    @Resource(name="complexLogicHandler")
    def complexLogicHandler

    def wrapLogic() {
        complexLogicHandler.handleLogic()
    }
}

```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Packaging plugins

Plugin types may be packaged in two different ways to upload to MWS:

- A simple Groovy file containing a single plugin type definition.
- A JAR file containing one or more plugin types, translators, and custom components.

While each may be uploaded to MWS using the REST API or the User Interface as described in [Add or update plugin types on page 1739](#), using a JAR file is recommended. Using a simple Groovy file is useful for testing and generating proof of concept work, but does not allow the use of several features of plugins.

The principles of packaging a plugin type or set of plugin types in a JAR file are very simple. Simply compile the classes and package in a typical JAR file. All classes ending in "Plugin" are automatically attempted to be loaded as a plugin type, all classes ending in "Translator" are attempted to be loaded as a translator, and all classes annotated as a custom component will be attempted to be loaded. It is recommended that a build framework is used to help with compiling and packaging the JAR file, such as

[Gradle](#). This makes it easy to declare a dependency on the necessary JAR files used in plugin development and to debug, compile, and test plugin code.

In addition to using utility services such as translators, packaging plugin types in JAR files allows the creation of a single project for multiple related plugin types and bundling of external dependencies. These two features are discussed in the following sections.

## Plugin projects and metadata

Each plugin type has information attached to it, called metadata, which describes the origin and purpose of the plugin type. Additionally, a JAR file may also contain a project file which defines default metadata attributes for all plugin types in the JAR. Initial plugins, or plugins that will be created on loading of the JAR file if they do not exist, are also able to be defined on a project file. In all cases, metadata declared on a plugin type will override the metadata defined on the project file.

To define a project file, simply add a class to JAR file that ends in "Project." This file will attempted to be loaded as the project file. Every field on a project file, and even the file itself, is optional. All available fields are shown in the example below.

```

class SampleProject {
    // Plugin information
    String title = "Sample"
    String description = "Sample plugin types"
    String author = "Our Company."
    String website = "http://example.com"
    String email = "sample@example.com"
    Integer eventComponent = 1
    // Versioning properties
    String version = "0.1"
    String mwsVersion = "7.1 > *"
    String commonsVersion = "0.9 > *"
    String license = "APACHE"

    // Documentation properties
    String issueManagementLink = "http://example.com/ticket-system/sample-plugins"
    String documentationLink = "http://example.com/docs/sample-plugins"
    String scmLink = "http://example.com/git/sample-plugins"

    // Plugins that are to be created with these properties only when they do NOT exist
    // This does not override any existing plugin instance configuration
    def initialPlugins = {
        /*
         * Multiple instances of plugins may be defined here.
         * In this case, 'sample' is the id of the plugin
         */
        sample {
            pluginType = "Sample"
            // All properties except for "pluginType" are optional
            pollInterval = 30
            autoStart = true
            // Although it is possible to set plugin precedence, it is not recommended
            // may already be taken and plugin creation will fail in this case
            precedence = 5
            config {
                configParam = "value"
            }
        }
        // Another plugin with an ID of 'sample2'
        sample2 {
            ...
        }
    }
}

```

As can be seen, metadata information about the plugin type(s), versions, and documentation are available. These are displayed when viewing plugin information in the User Interface or through the REST API.

Any of these properties except for `initialPlugins`, `mwsVersion`, and `commonsVersion` may be overwritten by the plugin type class itself by using static properties. A simple example is shown below.

```

package example

class SamplePlugin {
    // Properties may be typed, untyped, final, or otherwise,
    // but they MUST be static
    static version = "0.2"
    static title = "Sample plugin"
    static description = "This sample plugin is used to demonstrate metadata information"
    static author = "Separate Division"
    static eventComponent = 1

    ... // Rest of the plugin type definition
}

```

### Event component

The `eventComponent` field is explored in [Creating events and notifications on page 1704](#).

### MWS and commons versions

The `mwsVersion` and `commonsVersion` fields are used to restrict the versions of MWS and plugin framework with which the plugin project may be used. Each field is of the format `FIRST_VERSION > LAST_VERSION`, where `FIRST_VERSION` is the first supported MWS or plugin framework version (inclusive), and `LAST_VERSION` is the last supported MWS or plugin framework version (inclusive). Each version must take the format of `##` or `##.##`, as in `7.1`, or `7.1.2`. An asterisk (`*`) is used to denote any version, and may be used for the first or the last version.

Although support for restricting both the MWS and commons versions are provided, it is recommended to use the commons version restriction always and the MWS version restriction where necessary. Restrictions on the commons version prevent plugin loading errors while restrictions on the MWS version prevent runtime errors such as missing support for certain MWS API versions.

Typically the `mwsVersion` and `commonsVersion` fields are set as shown above, with the first version set to a specific number, and the last version set to any (an asterisk). This is the recommended approach for setting both fields. It is not recommended to use any version (asterisk) for the first version. Some examples of `mwsVersion` and `commonsVersion` values are shown below with explanations of how they behave.

```

String mwsVersion = "7.1 > *" // Any MWS version 7.1.0 and greater is supported
(including 7.2, etc)
String mwsVersion = "7.1.3 > *" // Any MWS version 7.1.3 and greater is supported
(including 7.2, etc)
String mwsVersion = "7.1 > 7.1.3" // Any MWS version between 7.1.0 and 7.1.3 is
supported
String mwsVersion = "*" > "*" // Any MWS version is supported (not recommended!)
String mwsVersion = "*" > 7.2" // Any MWS version up to 7.2 is supported (not
recommended!)

String commonsVersion = "0.9 > *" // Any framework version 0.9.0 and greater is
supported (including 1.0, etc)
String commonsVersion = "0.9.3 > *" // Any framework version 0.9.3 and greater is
supported (including 1.0, etc)
String commonsVersion = "0.9 > 0.9.3" // Any framework version between 0.9.0 and 0.9.3
is supported
String commonsVersion = "*" > "*" // Any framework version is supported (not
recommended!)
String commonsVersion = "*" > 1.0" // Any framework version up to 1.0 is supported (not
recommended!)

```

If the `mwsVersion` or `commonsVersion` fields are formatted incorrectly, the plugin project will fail to load. If a plugin project is uploaded to MWS and the version check fails, the project will fail to load with an error message about the `mwsVersion` or `commonsVersion`.

 The `mwsVersion` and `commonsVersion` fields cannot be overridden by a single plugin type, but can be set only at the plugin project level. This prevents mixing of MWS and commons version requirements within a single project.

### Initial plugins

The initial plugins closure provides the flexibility to insert plugin instances when the JAR is loaded. This occurs at two points: when the plugin JAR is first uploaded to MWS, and when MWS is restarted. As shown in the example above, the ID, `pluginType`, and other properties may be configured for multiple plugins.

The nature of Groovy closures means that programmatic definition of initial plugins is possible. This may even be based on the MWS application configuration. Two properties are automatically available in the `initialPlugins` closure:

- `appConfig` – Contains the MWS application configuration. Any configuration parameter is available for access as documented on [Configuration on page 1772](#).
- `suite` – Contains the currently configured suite that MWS is running in. This is equivalent to the `mws.suite` configuration parameter, and is an instance of [Suite](#).

### Native plugin case study

The Native JAR file utilizes many of the features discussed above. In the root of the JAR file, a compiled class called `NativeProject` exists which defines all of the metadata fields, including `initialPlugins`. Trying to create an initial plugin presents two distinct problems:

- The plugin should be initialized only if the suite is CLOUD.
- The plugin type configuration must contain an entry referencing the configured `mws.home.location` parameter, or the configured `MWS_HOME` location.

The `initialPlugins` closure is defined as follows:

```

import com.adaptc.mws.plugins.Suite

class NativeProject {
  ... // Metadata fields

  def initialPlugins = {
    // Initialize the cloud-native plugin only if the suite is CLOUD
    if (suite==Suite.CLOUD) {
      'cloud-native' {
        pluginType = "Native"
        pollInterval = 30
        config {
          // Use the appConfig property to retrieve the current MWS HOME
          getCluster = "file://${appConfig.mws.home.location}/etc/nodes
        }
      }
    }
  }
}

```

## Managing external dependencies

External dependencies (e.g. JAR files) may be included and referenced in JAR files. Certain rules must also be followed in order to have the dependencies loaded from the JAR file correctly:

The plugin type must bundle all external dependency JARs in the root of the plugin type JAR file.

An entry must be included in the `MANIFEST.MF` file that references each of these bundled JAR files as a space separated list:

```
Class-Path: dependency1.jar dependency2.jar dependency3.jar
```

Assuming that these rules are followed and that the plugin type is uploaded using the REST API or the User Interface, the dependent JARs will first be loaded and then the new plugin type and associated files will be loaded.

## Documenting plugin types

Documentation may also be included in JAR files by placing one or more [Markdown](#) formatted files in the root of the project JAR file. These files will be processed dynamically by MWS and presented as documentation pages for the respective plugin types within the MWS plugin user interface pages. Markdown is a simple text-to-HTML format used in some of the most popular open-source repositories such as [GitHub](#) and [BitBucket](#). To help provide plugin developers use a single place or file for documentation, the conventional use of "README.md" as documentation was followed within MWS.

### Documentation file naming

Each documentation filename must start with "README" and end with ".md". If only one documentation file is needed for bundled plugin type(s), it is recommended to call the file "README.md". For multiple plugin types, the file name must contain the plugin type name without the "Plugin" suffix in the format of "README-<PluginName>.md". For example, if a plugin project JAR file contained the plugin type classes "MyPlugin", "ABTestPlugin", and "ImportantPlugin", the documentation files would be located in the root of the JAR file and would be called "README-My.md", "README-ABTest.md", and "README-Important.md"

respectively. If a "README" file does not exist for a certain plugin type, the main "README.md" file (if provided) will be used as documentation for that plugin type.

### Markdown syntax

The Markdown syntax supported by MWS is very close to [GitHub Flavored Markdown](#). Internally, the [pegdown](#) Markdown processor is used to generate the HTML with the TABLES, ABBREVIATIONS, FENCED\_CODE\_BLOCKS, SMARTYPANTS, DEFINITIONS, and QUOTES extensions enabled. HTML tags may also be used directly in order to create more refined formatting of the documentation, but this is discouraged with the exception of inserting the configuration reference table discussed below.

For example, the TABLES extension may be used to easily create HTML tables:

```
Name | Notes
-----|-----
Bob   | Knows how to use MWS plugins but has never created one
George | Writes MWS plugins in his spare time
```

The only main difference from standard Markdown processors is that block quotes (marked by lines prepended with '> ') are shown as highlighted information boxes when displayed in MWS. This may be used to draw more attention to informational or warning messages without writing custom HTML.

```
> Warning: The use of this plugin type requires that MWS and MWM are configured
correctly as described in
> the MWS user guide.
```

### Configuration reference table

A table of available configuration parameters is often constructed in documentation for each plugin type. To ease the burden on the plugin developer of maintaining this documentation and the constraints on the plugin type, a table generated from the constraints (see [Configuration constraints on page 1686](#)) and included messages is available by using the following HTML in the README file(s):

```
<div class="configuration-table">This section will be replaced by MWS with the
configuration parameters table</div>
```

The text within the div container may be anything, but should state something helpful such as that it is placeholder in cases where the documentation may be viewed within other contexts such as on GitHub.

The generated table includes the following columns for each configuration parameter listed in the constraints: name, key, required, type, description. The "name" and "description" values are retrieved from the "help" and "label" messages bundled in the plugin JAR (see the labels and help messages section in [Configuration constraints on page 1686](#) for more information).

### Web services reference sections

Documentation for exposed web services (see [Exposing web services on page 1696](#)) is also able to be generated automatically. Instead of a single table as done with configuration parameters, a section with several tables (possible URL access points, URL parameters, and response fields) and additional information is generated for each exposed web service. This is available by using the following HTML in the README file(s):

```
<div class="webservice-sections">This section will be replaced by MWS with the web
service documentation</div>
```

The text within the `div` container may be anything, but should state something helpful such as that it is placeholder in cases where the documentation may be viewed within other contexts such as on GitHub.

### Changing heading sizes

The generated sections each begin with an `<h2>` heading with the name of the web service. If a different heading size (`h3`, `h4`, etc.) is desired, this may be done with the following HTML:

```
<div class="webservice-sections" data-level="3">This section will be replaced by MWS
with the web service documentation</div>
```

Notice the `data-level` attribute, which contains the number used in the HTML `h` tag.

### Message codes

Just as with the configuration table, the data for the content is generated automatically from the web service method name and from `i18n` messages (see [i18n messaging on page 1683](#)) bundled in the plugin JAR file. Message codes are available to customize the label and description of the web service. Codes are also available to define an arbitrary number of URL parameters and response fields. These do not need to be defined, but are helpful. The following table defines each message used in generating the documentation for web services.

Name	Message code	Description
<b>Web Service Label</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.label</code>	The label used as the heading for the section, defaults to the naturally capitalized method name if not present.
<b>Web Service Description</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.help</code>	Paragraph text describing the web service and its functionality, outputs, etc.
<b>Parameter Key</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.parameter&lt;n&gt;.key</code>	The <code>n</code> th URL parameter, starting at 1 (example: <code>id</code> ).
<b>Parameter Label</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.parameter&lt;n&gt;.label</code>	The label for the <code>n</code> th URL parameter, defaults to the naturally capitalized key if not present.

Name	Message code	Description
<b>Parameter Type</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.parameter&lt;n&gt;.type</code>	The type for the <i>n</i> th URL parameter, defaults to <i>String</i> if not present.
<b>Parameter Description</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.parameter&lt;n&gt;.help</code>	The description or help text for the <i>n</i> th URL parameter.
<b>Response Field Key</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.return&lt;n&gt;.key</code>	The <i>n</i> th response field, starting at 1 (example: <i>success</i> ).
<b>Response Field Label</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.return&lt;n&gt;.label</code>	The label for the <i>n</i> th response field, defaults to the naturally capitalized key if not present.
<b>Response Field Type</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.return&lt;n&gt;.type</code>	The type for the <i>n</i> th response field, defaults to <i>String</i> if not present.
<b>Response Field Description</b>	<code>&lt;pluginType&gt;.webServices.&lt;webServiceMethod&gt;.return&lt;n&gt;.help</code>	The description or help text for the <i>n</i> th response field.

As an example, suppose that a web service method called "doSomething" exists on a plugin type named "MyExamplePlugin". This web service expects two URL parameters: *id*, an integer, and *action*, a string. The response body consists of a JSON object with two fields: *success*, a boolean value, and *messages*, a list of strings. The following messages would serve to generate helpful documentation:

```

messages.properties
-----

# web service messages
myExamplePlugin.webServices.doSomething.label=Do Something Important
myExamplePlugin.webServices.doSomething.help=This web service does something important
with the input parameters.
# parameters
myExamplePlugin.webServices.doSomething.parameter1.key=id
myExamplePlugin.webServices.doSomething.parameter1.label=ID
myExamplePlugin.webServices.doSomething.parameter1.type=Integer
myExamplePlugin.webServices.doSomething.parameter1.help=The identifier of an object
myExamplePlugin.webServices.doSomething.parameter2.key=action
myExamplePlugin.webServices.doSomething.parameter2.label=Action # same as the default
would be
myExamplePlugin.webServices.doSomething.parameter2.type=String # same as the default
would be
myExamplePlugin.webServices.doSomething.parameter2.help=The action to perform

# response fields
myExamplePlugin.webServices.doSomething.return1.key=success
myExamplePlugin.webServices.doSomething.return1.label=Success # same as the default
would be
myExamplePlugin.webServices.doSomething.return1.type=Boolean
myExamplePlugin.webServices.doSomething.return1.help=True if the request succeeded,
false otherwise
myExamplePlugin.webServices.doSomething.return1.key=messages
myExamplePlugin.webServices.doSomething.return1.label=Error Messages
myExamplePlugin.webServices.doSomething.return1.type=List of Strings
myExamplePlugin.webServices.doSomething.return1.help=Error messages describing the
reason why success is false.

```

Note that if the *first* URL parameter key is `id`, the listed resource URLs will include the optional URL with the `id` parameter inline, such as `/rest/plugins/<pluginId>/services/<webService>/<id>`. Therefore, it is recommended to use `id` as parameter 1 if the web service expects a parameter with that key.

#### Related topics

- [Plugin developer's guide on page 1679](#)

## Example plugin types

Several plugin types are provided by Adaptive Computing for use in Moab Web Services. Examples of these include the Native and vCenter plugin types.

A sample plugin type in Groovy would resemble the following:

```

package sample

import com.adaptc.mws.plugins.*

class SamplePlugin extends AbstractPlugin {
    static author = "Adaptive Computing"
    static description = "A simple plugin in groovy"
    static version = "0.1"

    INodeRMService nodeRMService

    public void configure() throws InvalidPluginConfigurationException {
        def myConfig = config // "config" is equivalent to getConfig() in groovy
        def errors = []
        if (!myConfig.arbitraryKey)
            errors << "Missing arbitraryKey!"
        if (errors)
            throw new InvalidPluginConfigurationException(errors)
    }

    public void poll() {
        NodeReport node = new NodeReport("node1")
        node.resources.RES1.total = 5
        node.resources.RES1.available = 5
        node.state = NodeReportState.IDLE
        nodeRMService.save([node])
    }

    // Access at /rest/plugins/<id>/services/example-service
    public def exampleService(Map params) {
        return [success:true]
    }
}

```

### Related topics

- [Plugin developer's guide on page 1679](#)

## Moab Workload Manager resource manager integration

Moab Workload Manager possesses the concept of Resource Managers (RMs). While plugins can be related to RMs, they often provide greater functionality and serve more purposes than a typical RM. MWS must be represented in Moab as a RM to enable certain plugin features such as state reporting and handling RM events. This section describes the process of configuring Moab and additional details of its queries to MWS. It includes the following topics:

- [Configuring Moab Workload Manager on page 1731](#)
- [Resource manager queries on page 1735](#)

### Related topics

- [About Moab Web Services plugins on page 1672](#)

## Configuring Moab Workload Manager

During each iteration of Moab Workload Manager's cycle, it will query MWS through the RM interface to access current node, virtual machine, and job information. At this point, all reports are loaded from the database and consolidated into a single report of each object as explained in [Data consolidation on page 1677](#).

**i** All unset (or null) values for properties on reports are ignored.

In some cases it may be desired to query MWS directly for the current consolidated node, storage, virtual machine, and job reports. This may be done using the following URLs which return data in a format that is a subset of the API version 3 interface for each object (i.e. `/rest/nodes?api-version=3`, `/rest/vms?api-version=3`, `/rest/jobs?api-version=3`).

Query	Description
<code>/rest/plugins/all/rm/cluster-query?api-version=3</code>	Retrieves consolidated node, storage, and virtual machine reports from all plugins.
<code>/rest/plugins/&lt;ID&gt;/rm/cluster-query?api-version=3</code>	Retrieves consolidated node, storage, and virtual machine reports for the specified plugin ID.
<code>/rest/plugins/all/rm/workload-query?api-version=3</code>	Retrieves consolidated job reports from all plugins.
<code>/rest/plugins/&lt;ID&gt;/rm/workload-query?api-version=3</code>	Retrieves consolidated job reports for the specified plugin ID.

These queries have no effect on the data itself. In other words, reports are not removed or manipulated when RM queries are performed. These are manipulated only the RM services as described in [Reporting state data on page 1698](#).

### Examples

The following example uses cURL (see [cURL samples on page 1771](#)) to perform the query.

```

$ curl -u moab-admin:changeme! http://localhost:8080/mws/rest/plugins/all/rm/cluster-
query?api-version=3&pretty=true
{
  "nodes": {
    "n1.test": {
      "states": {
        "state": "IDLE"
      },
      "lastUpdatedDate": 1382386344,
      "resources": {
        "processors": {
          "configured": 4
        },
        "memory": {
          "configured": 8191,
          "available": 7206
        },
        "gres1": {
          "configured": 100
        }
      },
      "metrics": {
        "cpuLoad": 0.008233333333333334,
        "vmcount": 0,
        "cpuUtilization": 0.20083333333333333
      },
      "featuresReported": [
        "feature1"
      ],
      "ipAddress": "10.0.8.69",
      "operatingSystem": {
        "hypervisorType": "esx",
        "image": "vcenter-vcenter-esx-4.x",
        "virtualMachineImages": [
          "centos6-v7"
        ]
      },
      "variables": {
        "VCENTER_DATASTORE_REMOTE1": "datastore-448",
        "VCENTER_DATASTORE_LOCAL1": "datastore-411"
      },
      "attributes": {
        "MOAB_DATACENTER": {
          "value": "vcenter-datacenter-401",
          "displayValue": "vcenter-vcenter - adaptive data center"
        }
      }
    },
    "n2.test": {
      "states": {
        "state": "IDLE"
      },
      "lastUpdatedDate": 1382386344,
      "resources": {
        "processors": {
          "configured": 4
        },
        "memory": {
          "configured": 10239,
          "available": 9227
        },
        "gres1": {

```

```

        "configured": 100
    }
},
"metrics": {
    "cpuLoad": 0.00805,
    "vmcount": 0,
    "cpuUtilization": 0.19666666666666666
},
"featuresReported": [
    "feature1",
    "feature2"
],
"ipAddress": "10.0.8.76",
"operatingSystem": {
    "hypervisorType": "esx",
    "image": "vcenter-vcenter-esx-5.0",
    "virtualMachineImages": [
        "centos6-v7",
        "centos6",
        "win2008"
    ]
},
"variables": {
    "VCENTER_DATASTORE_REMOTE1": "datastore-448",
    "VCENTER_DATASTORE_LOCAL1": "datastore-415"
},
"attributes": {
    "MOAB_DATACENTER": {
        "value": "vcenter-datacenter-401",
        "displayValue": "vcenter-vcenter - adaptive data center"
    }
}
},
"n3.test": {
    "states": {
        "state": "IDLE"
    },
    "lastUpdatedDate": 1382386344,
    "resources": {
        "processors": {
            "configured": 4
        },
        "memory": {
            "configured": 10239,
            "available": 9229
        },
        "gres1": {
            "configured": 100
        }
    },
    "metrics": {
        "cpuLoad": 0.0097,
        "vmcount": 0,
        "cpuUtilization": 0.2375
    },
    "featuresReported": [
        "feature1"
    ],
    "ipAddress": "10.0.8.72",
    "operatingSystem": {
        "hypervisorType": "esx",
        "image": "vcenter-vcenter-esx-5.0",

```

```

        "virtualMachineImages": [
            "centos6-v7",
            "centos6",

            "win2008"
        ]
    },
    "variables": {
        "VCENTER_DATASTORE_REMOTE1": "datastore-448",
        "VCENTER_DATASTORE_LOCAL1": "datastore-416"
    },
    "attributes": {
        "MOAB_DATACENTER": {
            "value": "vcenter-datacenter-401",
            "displayValue": "vcenter-vcenter - adaptive data center"
        }
    }
},
"n4.test": {
    "states": {
        "state": "IDLE"
    },
    "lastUpdatedDate": 1382386344,
    "resources": {
        "processors": {
            "configured": 4
        },
        "memory": {
            "configured": 10239,
            "available": 9229
        },
        "gres1": {
            "configured": 100
        }
    },
    "metrics": {
        "cpuLoad": 0.007883333333333334,
        "vmcount": 0,
        "cpuUtilization": 0.1925
    },
    "featuresReported": [
        "feature2"
    ],
    "ipAddress": "10.0.8.77",
    "operatingSystem": {
        "hypervisorType": "esx",
        "image": "vcenter-vcenter-esx-5.0",
        "virtualMachineImages": [
            "centos6-v7",
            "centos6",
            "win2008"
        ]
    },
    "variables": {
        "VCENTER_DATASTORE_REMOTE1": "datastore-448",
        "VCENTER_DATASTORE_LOCAL1": "datastore-958"
    },
    "attributes": {
        "MOAB_DATACENTER": {
            "value": "vcenter-datacenter-401",
            "displayValue": "vcenter-vcenter - adaptive data center"
        }
    }
}

```

```

    }
  },
  "vms": {
    "vml": {
      "states": {
        "state": "DOWN",
        "powerState": "OFF"
      },
      "host": {
        "name": "nl.test"
      },
      "lastUpdatedDate": 1382386344,
      "resources": {
        "processors": {
          "configured": 4
        },
        "memory": {
          "configured": 12288
        }
      },
      "metrics": {
        "vmcount": 1
      }
    }
  },
  "storage": {}
}

```

### Related topics

- [Moab Workload Manager resource manager integration on page 1730](#)
- [Resource manager queries on page 1735](#)

## Resource manager queries

Moab Workload Manager must be configured to use MWS as a resource manager. Do the following:

- First, the following lines must be in the Moab Workload Manager configuration file or one of its included files:

```

RMCFG [mws]                TYPE=MWS
RMCFG [mws]                FLAGS=UserSpaceIsSeparate
RMCFG [mws]                BASEURL=http://localhost:8080/mws

```

*The **BASEURL** must match the configured URL of MWS.*

- The next step is to edit the MWS credential information in the Moab private configuration file (/opt/moab/etc/moab-private.cfg, by default). Here are the default values:

```

CLIENTCFG [RM:mws] USERNAME=moab-admin PASSWORD=changeme!

```

**i** **USERNAME** and **PASSWORD** must match the values of `auth.defaultUser.username` and `auth.defaultUser.password`, respectively, found in the MWS configuration file. The MWS RM contacts MWS directly using the base URL, username, and password configured.

Optionally, the **USERNAME** and **PASSWORD** configuration values may be specified directly in the Moab configuration file, though this is not recommended. Likewise, the **BASEURL** configuration value can be specified in the Moab private configuration file.

- Lastly, to enable such actions as submitting jobs as different users, the **ENABLEPROXY=TRUE** option must be present in the **ADMINCFG** configuration line, and the **OSCREDLLOOKUP** option must be set to **NEVER**, as follows:

```
ADMINCFG [1]           USERS=root           ENABLEPROXY=TRUE
OSCREDLLOOKUP        NEVER
```

- You may also want to configure SSL by using the following options (in either the **RMCFG** or **CLIENTCFG** section):
  - **SSLCACERT**: Lets you specify the absolute path to your SSL CA certificate. (This also enables the use of self-signed certificates, if desired.) It is recommended that you set this option in the Moab private configuration file. For example:

```
CLIENTCFG [RM:mws]   SSLCACERT=/path/to/cert.pem
```

- **SSLNOHOSTCHECK**: Lets you disable the SSL check to make sure that the actual server name matches the certificate's server name. For example:

```
#In moab-private.cfg
CLIENTCFG [RM:mws]   SSLNOHOSTCHECK=TRUE

#Or in moab.cfg
RMCFG [mws]         SSLNOHOSTCHECK=TRUE
```

 **WARNING:** This setting could compromise the security of the system and should not be used in production environments.

- **SSLNOPEERCHECK**: Lets you disable the SSL check to make sure that the certificate is valid.

```
#In moab-private.cfg
CLIENTCFG [RM:mws]   SSLNOPEERCHECK=TRUE

#Or in moab.cfg
RMCFG [mws]         SSLNOPEERCHECK=TRUE
```

 **WARNING:** This setting could compromise the security of the system and should not be used in production environments.

Related topics

- [Moab Workload Manager resource manager integration on page 1730](#)
- [Configuring Moab Workload Manager on page 1731](#)

## Plugin type management

Plugin types may be managed and accessed with Moab Web Services dynamically, even while running. Operations are provided to upload (add or update) plugin types and to list or show current plugin types. The available fields that are displayed with plugin types are given in [Fields: Plugin Types on page 2204](#). For more information on how these fields are set, see [Plugin projects and metadata on page 1721](#).



Plugin Type JAR or groovy files should never be manually copied into the `MWS_HOME/plugins` directory. They must be managed using the methods shown in this section or through the REST API (see [Plugin types on page 1607](#)).

Bundled plugin types are included automatically in Moab Web Services releases and may be utilized immediately after startup. See [Plugin management on page 1742](#) for more information on how to utilize these plugin types.



The plugin type documentation is now located in the plugin type management pages. See [Plugin type documentation on page 1738](#) for more information.

This section contains these topics:

- [Listing plugin types on page 1737](#)
- [Displaying plugin types on page 1738](#)
- [Plugin type documentation on page 1738](#)
- [Add or update plugin types on page 1739](#)

Related topics

- [About Moab Web Services plugins on page 1672](#)

### Listing plugin types

To list all plugin types, browse to the MWS home page (for example, <https://servername/mws>). Log in as the admin user, then click **Plugins > Plugin Types**.



#### Plugin Type List

This list shows all the plugin types that are available in Moab Web Services.

Add or Update Plugin Type

ID	Title	Author	Version	Has Poll Method	Instances
MSM	Moab Services Manager (MSM)	<a href="#">Adaptive Computing Enterprises, Inc.</a>	0.2	Yes	0
Native	Native	<a href="#">Adaptive Computing Enterprises, Inc.</a>	0.2	Yes	1

Related topics

- [Plugin type management on page 1737](#)

## Displaying plugin types

To show information about a plugin type, go to the **Plugin Type List** page and click the desired plugin type.

The screenshot shows the 'Show Plugin Type' interface for a 'Native' plugin. It includes a globe icon and the title 'Show Plugin Type'. The main content area lists various attributes: ID (Native), Title (Native), Description (Basic implementation of a native plugin), Author (Adaptive Computing Enterprises, Inc.), Email (mws.plugins@adaptivecomputing.com), License (APACHE), Version (0.2), MWS Version (7.1 > \*), Issues, Documentation, Sources, Has Poll Method (Yes), and Web Services (Name). Below the Web Services section, a message states 'There are no entries at this time'. The 'Default Configuration' section contains a table with columns for ID, Poll Interval, and Auto Start, showing a single entry for 'cloud-native' with a poll interval of 30. The 'Plugins' section contains a table with columns for ID and State, showing a single entry for 'cloud-native' with a state of 'Started'. At the bottom, there are two buttons: 'Add Plugin' and 'Plugin Type List'.

**Native**

**Native**

**Basic implementation of a native plugin**

**Adaptive Computing Enterprises, Inc.**

[mws.plugins@adaptivecomputing.com](mailto:mws.plugins@adaptivecomputing.com)

**APACHE**

**0.2**

**7.1 > \***

Issues

Documentation

Sources

**Yes**

Web Services

Name
There are no entries at this time

Default Configuration

ID	Poll Interval	Auto Start
cloud-native	30	

Plugins

ID	State
cloud-native	Started

Add Plugin

Plugin Type List

Related topics

- [Plugin type management on page 1737](#)

## Plugin type documentation

To show the documentation for a plugin type, go to the **Plugin Type List** page and click the desired plugin type. Then, click the **Open Documentation** button. This will display any documentation bundled with the plugin type.

Related topics

- [Plugin type management on page 1737](#)

## Add or update plugin types

Plugin types can be uploaded into Moab Web Services using a Groovy file, a Java Archive ([JAR](#)) file, or pasted Groovy code. To access the plugin type upload page, navigate to the **Plugin Type List** page and click **Add or Update Plugin Type**. The default interface of this page enables the uploading of a single Groovy class file or a JAR file.

When a plugin type is updated, by default all corresponding plugins created from the plugin type will be recreated. If this behavior is not desired, clear the **Do you want to reload all plugins to use this new version?** checkbox before uploading the plugin type.

### [Single class file](#)

Groovy files containing a single plugin type may be uploaded at the `/mws/admin/plugin-types/create` URL.

**Create Plugin Type**

Do you want to reload all plugins to use this new version?

+ Add files... Start upload Cancel upload Clear files

Cancel

Type or Paste Code

If the upload failed or an error occurred during initialization of the plugin, an error message will be displayed.

**Create Plugin Type**

Plugin Type with ID 'NotAValidClass' is an invalid plugin: Class name does not end with Plugin

Do you want to reload all plugins to use this new version?

+ Add files... Start upload Cancel upload Clear files

sample.groovy	0.04 KB	<div style="width: 100%; height: 10px; background-color: orange;"></div>	<input type="button" value="Cancel"/>
---------------	---------	--------------------------------------------------------------------------	---------------------------------------

Cancel

Type or Paste Code

### [JAR file](#)

A JAR file, as described in [Packaging plugins on page 1720](#), containing one or more plugins may also be uploaded using the same process as the Groovy file.

Click **Add files...**, select the `.jar` file, and click the **Start upload** button. If the upload failed or an error occurred during initialization of the plugin(s), an error message will be displayed.

The JAR upload process differs from the single file in that if successful, the name of the JAR file itself is displayed instead of the plugin name(s).



### Create Plugin Type

Do you want to reload all plugins to use this new version?

+ Add files...			Start upload	Cancel upload	Clear files
samples.jar	28.16 KB				

Cancel

Type or Paste Code

### Code

To paste or type code directly into MWS and have it be loaded as a single class file, click **Type or Paste Code**, and type or paste the code into the presented text box.



## Create Plugin Type

Upload File(s)

Do you want to reload all plugins to use this new version?

Code

Save

Cancel



## Create Plugin Type

Upload File(s)

Do you want to reload all plugins to use this new version?

Code

```
package sample.polling;

import com.ace.mws.plugins.*

public class PollingPlugin extends AbstractPlugin {
    static final title = "Polling Sample"
    public void poll() {
```

Save

Cancel

When the code is in the box, click **Create**. If the upload succeeded and the code was able to be compiled as Groovy, the browser will be redirected to the **Show Plugin Type** page. If the upload failed or an error occurred during compilation or initialization of the plugin, an error message will be displayed.



You may need to refer to the MWS [log](#) file for additional details and error messages in the case of a failure.

### Related topics

- [Plugin type management on page 1737](#)

## Plugin management

Plugins may be managed and accessed with Moab Web Services dynamically, even while running. This includes plugin instance and lifecycle management. Additionally, default configuration values may be set for new plugins. In order to access custom web services, the REST API must be utilized as described in [Accessing Plugin Web Services on page 1605](#). The available fields that are displayed with plugins are given in the [Fields: Plugins](#) reference.

This section contains these topics:

- [Listing plugins on page 1742](#)
- [Creating a plugin on page 1742](#)
- [Displaying a plugin on page 1743](#)
- [Modifying a plugin on page 1744](#)
- [Deleting a plugin on page 1745](#)
- [Monitoring and lifecycle controls on page 1745](#)
- [Setting default plugin configuration on page 1747](#)

Related topics

- [About Moab Web Services plugins on page 1672](#)

### Listing plugins

To list all plugins, browse to the MWS home page (for example, <https://servername/mws>). Log in as the admin user, then click **Plugins > Plugins**.



#### Plugin List

This list shows all the plugins that have been configured in Moab Web Services.

[Add Plugin](#)

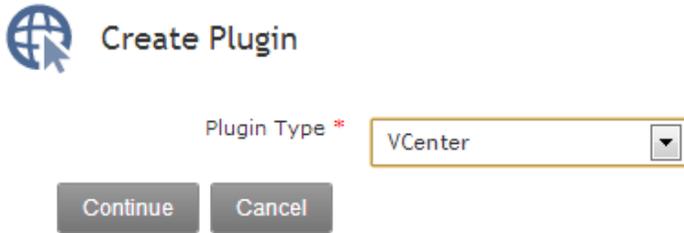
ID	Plugin Type	State	Poll Interval	Auto Start
cloud-native	Native	Started	30	Yes

Related topics

- [Plugin management on page 1742](#)

### Creating a plugin

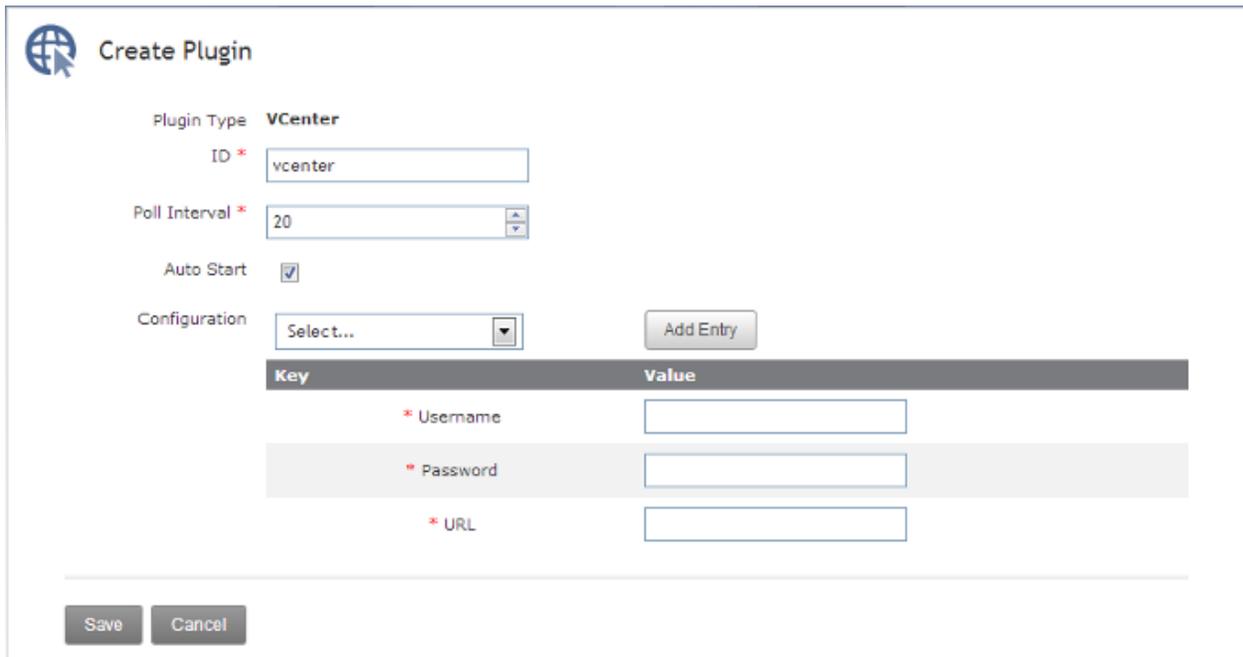
To create a plugin, go to the **Plugin List** page and click **Add Plugin**. First, a **Plugin Type** must be selected to continue to actually create the plugin.



**Create Plugin**

Plugin Type \*

The page is automatically built to support the plugin type's constraints (see [Configuration constraints on page 1686](#)). The **ID** field will be automatically filled in with a suggested value, and the **Poll Interval** field will be displayed only if the plugin type has a `poll` method. The required configuration fields are displayed by default, and optional fields may be selected and added to the configuration from the drop down at the top of the configuration section. See the [Fields: Plugins](#) reference section for more information on the fields.



**Create Plugin**

Plugin Type **VCenter**

ID \*

Poll Interval \*

Auto Start

Configuration

Key	Value
* Username	<input type="text"/>
* Password	<input type="text"/>
* URL	<input type="text"/>

Related topics

- [Plugin management on page 1742](#)

## Displaying a plugin

To show information about a plugin, go to the **Plugin List** page and click the desired plugin ID.



### Show Plugin

ID	cloud-native				
Plugin Type	Native				
State	Started				
Poll Interval	30				
Auto Start	Yes				
Configuration	<table><thead><tr><th>Key</th><th>Value</th></tr></thead><tbody><tr><td>Get Cluster</td><td>file:///opt/mws/etc/nodes.txt</td></tr></tbody></table>	Key	Value	Get Cluster	file:///opt/mws/etc/nodes.txt
Key	Value				
Get Cluster	file:///opt/mws/etc/nodes.txt				
Web Services	<table><thead><tr><th>Name</th></tr></thead><tbody><tr><td colspan="1">There are no entries at this time</td></tr></tbody></table>	Name	There are no entries at this time		
Name					
There are no entries at this time					

### Related topics

- [Plugin management on page 1742](#)

## Modifying a plugin

To modify a plugin, go to the **Plugin List** page, click the desired plugin ID, and then click **Edit**. See the [Fields: Plugins](#) reference section for more information on available fields.



### Edit Plugin

ID	cloud-native						
Plugin Type	Native						
State	Started						
Poll Interval *	<input type="text" value="30"/>						
Auto Start	<input checked="" type="checkbox"/>						
Configuration	<input type="text" value="Select..."/>						
	<input type="button" value="Add Entry"/>						
	<table><thead><tr><th>Key</th><th>Value</th><th></th></tr></thead><tbody><tr><td>Get Cluster</td><td><input type="text" value="file:///opt/mws/etc/nodes.txt"/></td><td><a href="#">Remove</a></td></tr></tbody></table>	Key	Value		Get Cluster	<input type="text" value="file:///opt/mws/etc/nodes.txt"/>	<a href="#">Remove</a>
Key	Value						
Get Cluster	<input type="text" value="file:///opt/mws/etc/nodes.txt"/>	<a href="#">Remove</a>					
	<input type="button" value="Update"/> <input type="button" value="Cancel"/>						

Related topics

- [Plugin management on page 1742](#)

## Deleting a plugin

To delete a plugin, go to the **Plugin List** page, click the desired plugin ID, and then click **Delete**. A confirmation message is shown. If the **OK** button is clicked, the plugin is deleted from the system and cannot be recovered, including all configuration.

Related topics

- [Plugin management on page 1742](#)

## Monitoring and lifecycle controls

To monitor and control the lifecycle of plugins, browse to the MWS home page (for example, <https://servername/mws>). Log in as the admin user, then click **Plugins > Plugin Monitoring**. This page displays the current state of all plugins as well as their polling status.



### Plugin Monitoring

This page monitors the status of all plugins in Moab Web Services.

Tuesday, June 12, 2012

11:28:11 AM

Reload when poll occurs

#### Active Plugins

ID	Plugin Type	Last Poll	Next Poll	Actions
cloud-native	Native	00:00:08	00:00:21	  
no-polling	Logging			 

#### Disabled Plugins

ID	Plugin Type	State	Actions
There are no Disabled Plugins set up at this time			

**i** If plugins are created from plugin types which do not have a `poll` method, their lifecycle controls will be limited. Any information below which mentions polling does not apply to the 'no-polling' plugin shown in the screenshots.

Active plugins

Active plugins are those which are in the Started or Paused states. These are available to receive events such as polling. If paused, a plugin will not receive events but is not actually stopped, therefore no stop events are triggered.

The following images demonstrate the status of plugins in the active states.

**Active Plugins**

ID	Plugin Type	Last Poll	Next Poll	Actions
cloud-native	Native	00:00:08	00:00:21	⏹️ ⏸️ ▶️
no-polling	Logging			⏹️ ⏸️

Started plugins which can include the relative time of the last poll as well as the time of the next poll in a countdown format. Action buttons are available to stop or pause the plugin as well as trigger an immediate poll event.

**Active Plugins**

ID	Plugin Type	Last Poll	Next Poll	Actions
cloud-native	Native	00:00:26		⏹️ ▶️ ▶️
no-polling	Logging			⏹️ ▶️

Paused plugins which can include only the last polling time. Action buttons are available to stop or resume the plugin, as well as trigger an immediate poll event.

Disabled plugins

Disabled plugins are those which are in the Stopped or Errored states. These plugins do not receive events such as polling. If errored, a plugin may either be stopped, which represents a "clearing" of the error, or started normally. However, if no action is taken on an errored plugin, it likely will not start due to the fact that most plugins are put into the errored state during startup of the plugin.

The following images demonstrate the representation of plugins in the disabled states.

**Disabled Plugins**

ID	Plugin Type	State	Actions
no-polling	Logging	Stopped	▶️

Stopped plugins. A single action button is available to attempt to start the plugin.

Disabled Plugins			
ID	Plugin Type	State	Actions
with-error	Logging	Errored	 

An errored plugin. As mentioned previously, action buttons are available to stop the plugin or clear the error as well as attempt to start the plugin. If the start fails, an error message will be displayed.

Related topics

- [Plugin management on page 1742](#)

## Setting default plugin configuration

Configuration of default values for plugin configuration parameters involves setting fields in the MWS configuration file. These values are used if no values are provided when creating a new plugin. Additionally, the default values will be displayed to the user on the **Create Plugin** page.

The parameters to configure are documented on [Configuration on page 1772](#) and comprise most values starting with plugins.

Related topics

- [Plugin management on page 1742](#)

## Plugin services

To use the built-in services, declare a variable with the correct name as a property in the plugin class.

The convention for each service name is to remove the leading "I" and lower case the resulting first letter. For example, the property to use the `IMoabRestService` would be called `moabRestService`. The following is an example of using the `IPluginControlService` in this manner.

```
Using the IPluginControlService
-----
package example;
import com.adaptc.mws.plugins.*;

class ExamplePlugin {
    IPluginControlService pluginControlService;

    public poll() {
        // Use service...
        pluginControlService.stop("pluginId");
    }
}
```

**i** Use of the Groovy anonymous type "def" may also be used. For example, the service definition above would use `def pluginControlService` instead of `IPluginControlService pluginControlService`.

**⚠** Do *not* attempt to create a new instance of the services before use, such as in a constructor. The services will be automatically injected before any methods are called on the plugin.

#### API documentation

The `com.adaptc.mws.plugins` package contains interfaces for all bundled services available to plugin types. These may be used as discussed above. All services begin with "I" and end with "Service", as in [IMoabRestService](#) ([Moab REST service on page 1748](#)).

#### Related topics

- [About Moab Web Services plugins on page 1672](#)

## Job RM service

The job RM service may be used to report job state data to Moab Workload Manager through the RM interface. See [Reporting state data on page 1698](#) for more information. It may also be used to retrieve previous reports made by a plugin. Please note that due to data consolidation (see [Data consolidation on page 1677](#)), old job reports may no longer exist in the database by the time the query is done.

The `jobRMService` property will be injected with a class of type [IJobRMService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

#### Related topics

- [Plugin services on page 1747](#)

## Moab REST service

The Moab REST service may be used to access the MWS RESTful API (see [Resources introduction on page 1446](#)) in plugins. All "requests" made through this service are internal only and no data is actually transmitted over the network. See [Accessing MWS REST resources on page 1702](#) for more information.

The `moabRestService` property will be injected with a class of type [IMoabRestService](#) in all plugin types.

#### Accessing resources

In order to access a resource, a relative URL matching that in the documentation must be used along with a HTTP method, such as GET, POST, PUT, or DELETE. The method names on [IMoabRestService](#) match the HTTP methods directly. For example, to call a GET operation on `/rest/jobs`, use the following code:

```
moabRestService.get("/rest/jobs")
```

### Using parameters correctly

Although the ordering of the parameters for each method on `IMoabRestService` may seem confusing at first glance, this is to allow for easy use with Groovy. Examples are given below for each combination of parameters:

```
String URL
-----
moabRestService.get("/rest/jobs")
```

```
Map options, String URL
-----
moabRestService.get("/rest/jobs", hooks:true, contentType:"application/json")
```

```
String URL, Closure data
-----
moabRestService.get("/rest/jobs/job.1") {
    [flags:"RESTARTABLE"]
}
```

```
Map options, String URL, Closure data
-----
moabRestService.get("/rest/jobs/job.1", hooks:true, contentType:"application/json") {
    [flags:"RESTARTABLE"]
}
```

### Options

The following options are valid in each method call supporting the `options` parameter:

Name	Type	Default	Description
<code>data</code>	See <a href="#">Valid data types on page 1750</a>	--	Specifies the body of the "request." This can be overwritten by the <code>data</code> Closure parameter.
<code>hooks</code>	Boolean	false	Specifies whether or not hooks (see <a href="#">Pre and post-processing hooks on page 1434</a> ) are run as part of the "request."
<code>contentType</code>	String	application/json	Indicates the content type used for the request.
<code>params</code>	Map	--	Indicates URL query parameters to use for the "request," such as <code>query</code> , <code>sort</code> , <code>proxy-user</code> , or others.

### Valid data types

If the `data Closure` parameter is specified, it overwrites the `data` option. In each case, there are four valid types for the `data` option or return value of the `data closure`:

- A non-null [JSON](#) instance.
- A valid JSON string. This will be converted into a [JSON](#) instance.
- A valid Map instance. This will be converted into a [JSONObject](#) instance.
- A valid List instance. This will be converted into a [JSONArray](#) instance.

 A [JSONException](#) may be thrown if the JSON string is invalid or the Map or List contains values that cannot be serialized to JSON.

### Related topics

- [Plugin services on page 1747](#)

## Node RM service

The node RM service may be used to report node state data to Moab Workload Manager through the RM interface. See [Reporting state data on page 1698](#) for more information. It may also be used to retrieve previous reports made by a plugin. Please note that due to data consolidation (see [Data consolidation on page 1677](#)), old node reports may no longer exist in the database by the time the query is done.

The `nodeRMService` property will be injected with a class of type [INodeRMService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

### Related topics

- [Plugin services on page 1747](#)

## Plugin control service

 This service is currently in Beta. Interfaces may change significantly in future releases.

The control service allows lifecycle management operations to be performed on plugins. It also provides methods to create and retrieve plugins. Note that the plugin control service may be used by other plugins, allowing one plugin to dynamically create, retrieve, start, or stop plugins.

The `pluginControlService` property will be injected with a class of type [IPluginControlService](#) in all plugin types.

Examples

Create plugin with default configuration

---

```
try {
    if (pluginControlService.createPlugin("myPlugin", "Native"))
        log.info "myPlugin was created successfully!"
    else
        log.warn "There was an error creating myPlugin"
} catch(PluginStartException e) {
    log.warn "There was a problem starting the new plugin: ${e.message}"
} catch(InvalidPluginConfigurationException e) {
    log.warn "There were errors with the plugin's configuration: ${e.errors}"
}
```

Create plugin with custom configuration

---

```
if (pluginControlService.createPlugin("myPlugin", "Native", [autoStart:false,
pollInterval:600]))
    log.info "myPlugin was created successfully!"
else
    log.warn "There was an error creating myPlugin"
```

Start plugin

---

```
try {
    pluginControlService.start("myPlugin")
} catch(PluginStartException e) {
    log.warn "There was a problem starting the plugin: ${e.message}"
} catch(InvalidPluginException) {
    log.warn "The plugin 'myPlugin' is invalid"
} catch(InvalidPluginConfigurationException e) {
    log.warn "The plugin has an invalid configuration: ${e.errors}"
}
```

Stop plugin

---

```
try {
    pluginControlService.stop("myPlugin")
} catch(PluginStopException e) {
    log.warn "There was a problem stopping the plugin: ${e.message}"
} catch(InvalidPluginException) {
    log.warn "The plugin 'myPlugin' is invalid"
}
```

Configure plugin

---

```
try {
    pluginControlService.configure("myPlugin")
} catch(InvalidPluginException) {
    log.warn "The plugin 'myPlugin' is invalid"
} catch(InvalidPluginConfigurationException e) {
    log.warn "The plugin has an invalid configuration: ${e.errors}"
}
```

## Related topics

- [Plugin services on page 1747](#)

## Plugin datastore service

The datastore service is provided to allow a plugin to persist data to the database that is isolated from all other persistent data. In other words, this service provides access to a plugin's individual datastore (see [Individual datastore on page 1694](#)).

The `pluginDatastoreService` property will be injected with a class of type [IPluginDatastoreService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

### Examples

Adding a single custom entry

```
package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def data = [:]
        ... // Add data here to the Map
        if (pluginDatastoreService.addData(collectionName, data))
            log.info("Data successfully added")
        else
            log.warn("There was an error adding the data")
    }
}
```

Adding multiple entries

```
package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def dataList = []
        dataList.add( /* Custom Map of data here */
        dataList << ... // Custom Map of data here
        if (pluginDatastoreService.addData(collectionName, dataList))
            log.info("Data entries successfully added")
        else
            log.warn("There was an error adding the data entries")
    }
}
```

## Updating a single entry

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def data = [:]
        ... // Add data here to the Map
        if (pluginDatastoreService.updateData(collectionName, "key", "value", data))
            log.info("Data successfully updated")
        else
            log.warn("There was an error updating the data")
    }
}

```

## Querying if a collection exists

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        if (pluginDatastoreService.exists(collectionName))
            log.info("Collection exists")
        else
            log.warn("The collection does not exist")
    }
}

```

## Querying contents of a collection

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def dataList = pluginDatastoreService.getCollection(collectionName)
        if (dataList!=null)
            log.info("Collection successfully queried")
        else
            log.warn("The collection does not exist!")
    }
}

```

## Retrieving a single entry

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def data = pluginDatastoreService.getData(collectionName, "key", "value")
        if (data!=null)
            log.info("Data successfully retrieved")
        else
            log.warn("The entry with key==value does not exist")
    }
}

```

## Removing a collection

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        def data = pluginDatastoreService.clearCollection(collectionName)
        // Data now contains the collection that was cleared
        if (data!=null)
            log.info("Collection successfully cleared")
        else
            log.warn("The collection does not exist!")
    }
}

```

## Removing a single entry

```

package example

public class ExamplePlugin {
    def pluginDatastoreService

    public void poll() {
        def collectionName = "collection1"
        if (pluginDatastoreService.removeData(collectionName, "key", "value"))
            log.info("Data entry successfully removed")
        else
            log.warn("The entry where key==value does not exist!")
    }
}

```

## Related topics

- [Plugin services on page 1747](#)

## Plugin event service

The event service is provided to ease the burden and reduce boilerplate code for creating new events and updating notification conditions. For more information on how to use this service, see [Creating events and notifications on page 1704](#).

The `pluginEventService` property will be injected with a class of type [IPluginEventService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

### Examples

#### Creating a custom event

```
package example

import com.adaptc.mws.plugins.IPluginEventService.Severity
import com.adaptc.mws.plugins.IPluginEventService.EscalationLevel
import com.adaptc.mws.plugins.IPluginEventService.AssociatedObject

public class ExamplePlugin {
    def pluginEventService

    public void poll() {
        // Create a completely custom event
        pluginEventService.createEvent(Severity.INFO, EscalationLevel.USER, 0x4F, "Custom
Type",
                                     "poll", "My event occurred", null, null)
    }
}
```

#### Creating a custom event with messages

```
package example

import com.adaptc.mws.plugins.IPluginEventService.Severity
import com.adaptc.mws.plugins.IPluginEventService.EscalationLevel
import com.adaptc.mws.plugins.IPluginEventService.AssociatedObject

public class ExamplePlugin {
    def pluginEventService

    public void poll() {
        // Use i18n messages for another event
        def args = ["arg1", "arg2"]
        pluginEventService.createEvent(Severity.WARN, EscalationLevel.POWER_USER, 0x5F,
"Custom Type",
                                     "poll", message
(code:"examplePlugin.customEvent.message", args:args), args,
                                     // AssociatedObjects or simple maps may be
used
                                     [new AssociatedObject(type:"type1", id:"id1"),
                                     [type:"type2", id:"id2"]])
    }
}
```

Creating an event from EventEnumeration

```

package example

import com.adaptc.mws.plugins.EventEnumeration
import com.adaptc.mws.plugins.IPluginEventService.Severity
import com.adaptc.mws.plugins.IPluginEventService.EscalationLevel
import com.adaptc.mws.plugins.IPluginEventService.AssociatedObject

public class ExamplePlugin {
    def pluginEventService

    public void poll() {
        // Messages are pulled for messages.properties file(s) and the arguments are used
        def args = ["arg1", "arg2"]
        pluginEventService.createEvent(MyEvents.EVENT_INFO, args, [[type:"type1", id:"id1]])
        pluginEventService.createEvent(MyEvents.EVENT_WARN, args, [[type:"type2", id:"id2]])
    }
}

@EventEnumeration
enum MyEvents {
    EVENT_INFO("Information", INFO, USER),
    EVENT_ERROR("Warning", WARN, USER)

    static final String EVENT_TYPE_PREFIX = "Example Plugin"
    static final String ORIGIN_SUFFIX = "poll"
}

```

Create or update a notification

```

package example

import com.adaptc.mws.plugins.IPluginEventService.EscalationLevel
import com.adaptc.mws.plugins.IPluginEventService.AssociatedObject

public class ExamplePlugin {
    def pluginEventService

    public void poll() {
        pluginEventService.updateNotification(EscalationLevel.POWER_USER, "There is an error
with node1",
// If non-null, this must always be an associated object, never null
new AssociatedObject(id:"node1", type:"node"), null)
    }
}

```

## Related topics

- [Events on page 1528](#)
- [Notifications on page 1585](#)
- [Notification conditions on page 1580](#)
- [Plugin services on page 1747](#)
- [Plugin developer's guide on page 1679](#)
- [Fields: Events on page 1894](#)

- [Resources introduction on page 1446](#)
- [Creating events and notifications on page 1704](#)

## SSL service

The SSL service may be used to manage and load certificates or keys from disk and create socket connections. See [Managing SSL connections on page 1714](#) for more information.

The `sslService` property will be injected with a class of type [ISslService](#) in all plugin types.

### Related topics

- [Plugin services on page 1747](#)

## Storage RM service

The storage RM service may be used to report storage state data to Moab Workload Manager through the RM interface. See [Reporting state data on page 1698](#) for more information. It may also be used to retrieve previous reports made by a plugin. Please note that due to data consolidation (see [Data consolidation on page 1677](#)), old storage reports may no longer exist in the database by the time the query is done.

The `storageRMService` property will be injected with a class of type [IStorageRMService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

### Related topics

- [Reporting state data on page 1698](#)
- [Plugin services on page 1747](#)

## Virtual machine RM service

The virtual machine RM service may be used to report virtual machine state data to Moab Workload Manager through the RM interface. See [Reporting state data on page 1698](#) for more information. It may also be used to retrieve previous reports made by a plugin. Please note that due to data consolidation [Data consolidation on page 1677](#), old virtual machine reports may no longer exist in the database by the time the query is done.

The `virtualMachineRMService` property will be injected with a class of type [IVirtualMachineRMService](#) in all plugin types. Note that it is not available for injection in translators or custom components.

### Related topics

- [Plugin services on page 1747](#)

# Plugin types

## Power Management Plugin

The Power Management plugin is used as a resource manager to Moab to report and manipulate the power state (On or Off) for each node. Moab considers nodes in the power state On or Off; however, through TORQUE and scripts, we are able to separate the Off state into those controlled through the operating system (Standby, Suspend, Hibernate, Shutdown) and those controlled through hardware (Off). This plugin provides an easy way to integrate with Moab to translate Moab's Off action into the desired TORQUE or script action for each node. A cluster will have multiple instances of this plugin when it has varied hardware integration and/or credentials.

### Creating a Power Management Plugin

To create a Power Management plugin, see [Creating a plugin on page 1742](#). During plugin creation, refer to the [Configuration on page 1758](#) section.

## Configuration

### Configuration Parameters

Name	Key	Required	Type	Description
Node Configuration File	nodeConfigurationFile	Yes	String	File containing list of nodes that use the scripts and credentials in this plugin instance. This is also the file to configure a particular node's off state, or an off state that will override the default off state for this instance.
Username File	usernameFile	Yes	String	File containing username issued to the scripts with the -u option.
Password File	passwordFile	Yes	String	File containing password issued to the scripts with the -p option.
Node Power Script	nodePowerScript	Yes	String	Script that powers on and off nodes and wakes them from a low power state.
Node Query Script	nodeQueryScript	Yes	String	Script that queries power state using an intelligent platform management interface.

Name	Key	Required	Type	Description
Default Power Off State	defaultPowerOffState	Yes	String	Actual state (Standby, Suspend, Hibernate, Shutdown, or Off) nodes will go into when Moab powers them off.
Max Threads	maxThreads	Yes	Integer	Thread count issued to the scripts with the -t option (defaults to 4).

## Plugin Management

For information on managing the IPMI plugin, including stopping it, starting it, and checking on its status, see the [Plugin management on page 1742](#) section of the MWS Guide.

## Web Services

### Node Power (Secured)

#### Resource URLs

Resource
/rest/plugins/<pluginId>/services/nodePower
/rest/plugins/<pluginId>/services/node-power

#### URL Parameters

Parameter	Name	Type	Description
nodes	Moab Nodes	String	A comma-delimited list of Moab node names. It is required.
power	The Power State	String	The power command Moab issues the node (On or Off).

#### Response Fields

Field	Name	Type	Description
success	Success Indicator	Boolean	True if the power script and/or <a href="#">pbsnodes on page 2705</a> was successful, otherwise false.

Field	Name	Type	Description
messages	Messages	List of Strings	Only present when the request was not successful or the node was not configured with the plugin instance. Contains error messages describing why the pbsnodes or the power script failed.

*Additional Information*

This web service was intended for Moab's use only and is exposed for debugging and testing your customized scripts.

## Reload Node Configuration (Secured)

*Resource URLs*

Resource
<code>/rest/plugins/&lt;pluginId&gt;/services/reloadNodeConfiguration</code>
<code>/rest/plugins/&lt;pluginId&gt;/services/reload-node-configuration</code>

*URL Parameters*

Parameter	Name	Type	Description
No URL parameters required			

*Response Fields*

Field	Name	Type	Description
success	Success Indicator	Boolean	True if the reload succeeded, otherwise false.
messages	Messages	List of Strings	Only present when the request failed. Contains error messages describing why the reload failed.

*Additional Information*

 The reloadNodeConfiguration web service must be run after any change to the node configuration file for it to take effect.

## Node Configuration File

The node configuration file is used when the plugin is first instantiated or the `reloadNodeConfiguration` web service is called. The plugin expects a file that is readable by the `tomcat` user and has a Moab node name on each line. If the user would like to override the default power-off state of the node, then the node name is followed by a space and the state. For example, a node configuration file might look like this:

```
node01.ac
node02.ac
node03.ac Hibernate
node04.ac Suspend
```

The valid power-off states include Standby, Suspend, Hibernate, Shutdown, and Off. If no power-off state is provided for the node in the configuration file, then the default power-off state will be used.

## The Node Power and Query Script

The plugin uses the power script to power on nodes from all power states and to power off nodes only into the Off power state. The plugin uses the power state of the node to decide whether to power on the node with `wake` or `on`. If the node is in Standby or Suspend, the plugin will call the script with the `wake` parameter. If the node is in Hibernate, Shutdown, or Off, the plugin will call the script with the `on` parameter. The plugin calls the power node script with the `off` parameter to put the node in the Off state (it uses TORQUE to put the node in the Standby, Suspend, Hibernate, and Shutdown state).

The plugin uses the query script to know if a node is in the Off power state. If the query script reports the node as Off, the plugin will report the node as Off to Moab. If the query script reports the node as On, the plugin will look to TORQUE to make sure the node is in a Running power state before it reports it as On.

The plugin passes the `usernameFile`, `passwordFile`, and `maxThreads` configuration parameters down to the scripts. The node power script is called with this syntax:

```
<nodePowerScript> -u <usernameFile> -p <passwordFile> -t <maxThreads> node01 node02
node03 ... <on|off|wake>
```

The node query script is called with this syntax:

```
<nodeQueryScript> -u <usernameFile> -p <passwordFile> -t <maxThreads> node01 node02
node03 ...
```

The plugin expects the scripts to print JSON to standard out. An example query script output would look like this:

```
[
  {
    "name": "node01.ac",
    "power": "ON",
    "Processor_2_Temp": 61,
    "Processor_1_Temp": 54
  },
  ...
]
```

Notice it is a list of nodes where each node has the required fields `name` and `power`. All the other key-value pairs will be reported to Moab as a generic resource as long as the value is a number.

The output for the node power plugin is not required; however, the output is read to give the user a detailed error message if needed. For both the node power and query scripts, if the field `error` exists, the plugin will log an error with all the strings in the list. An example error returned to the plugin would look like this:

```
[
  {
    "command": "ipmitool -I lan -H node01i -U admin -f /opt/moab/etc/power-
management/abc-plugin-password-file sdr type temperature",
    "name": "node01.ac",
    "error": [
      "big error"
    ]
  }
  ...
]
```

## Troubleshooting

The Power Management plugin logs all errors and warnings to the MWS log file, which is `/opt/mws/log/mws.log` by default. The `stacktrace.log` file, in the same directory as `mws.log`, can also be helpful in diagnosing problems. If your MWS supports notifications, they are also helpful in diagnosing the error states the plugin is in, if any. Just check for notifications from the `PowerManagement` plugin type and the instance that you are interested in. When the issue has been resolved, you can dismiss the notification. For more information, see the Notification and Notification Condition Resource in the MWS documentation.

### Set the appropriate MWS RM precedence

The Create/Edit Plugin pages give the option to set the precedence of the Moab RM plugin. The purpose of the Power Management Plugin is to report node power; however, if the precedence is too low another Moab RM plugin with a higher precedence and conflicting node might overwrite the node power. To check what MWS is reporting to Moab, go to the URL:

```
http://<MWS host>:8080/mws/rest/plugins/all/rm/cluster-query[?api-version=3]
```

To check what your plugin instance is reporting to Moab, use the URL:

```
http://<MWS host>:8080/mws/rest/plugins/<instance-name>/rm/cluster-query[?api-
version=3]
```

If the power is reported in your instance but not to Moab, please increase the precedence of the Moab RM plugin.

### Configure the MWS RM in Moab

First, the following lines must be in the Moab Workload Manager configuration file or one of its included files:

```

RMCFG [mws]          TYPE=MWS
RMCFG [mws]          FLAGS=UserSpaceIsSeparate
RMCFG [mws]          BASEURL=http://<mws host>:8080/mws

```

Next, edit the MWS credential information in the Moab private configuration file (`/opt/moab/etc/moab-private.cfg`, by default). Here are the default values:

```

CLIENTCFG [RM:mws]  USERNAME=moab-admin  PASSWORD=changeme!

```

For more information see the Resource manager queries section in the MWS documentation.

## Configure TORQUE with tomcat administrator

The plugin assumes that TORQUE is installed on the same host as MWS and that tomcat is an administrator. This can be verified with [qmgr on page 2726](#). Run the command:

```

qmgr -c 's s managers += tomcat@<mws_host>'

```

For more information see the Specifying non-root administrators section of the TORQUE documentation.

## Make sure the Node and Power scripts work first.

The default scripts are included in `/opt/moab/tools/mws/power_management` and have their own documentation with the `-h` option. They need to have a file that maps each node in the Moab cluster to the IPMI address that the script will need to call using `ipmitool`. It also needs a file that includes the IPMI password. After that is provided and `ipmitool` is installed and working, the scripts will successfully implement the interface needed for this plugin.

Related topics

- [pbsnodes -m](#)
- [Green computing overview](#)

# References

## Client code samples

The code samples contained in this section of the reference material are provided to help quick start integration with MWS. They are provided as a convenience and not as fully developed APIs.

All examples use the default configuration of MWS, including the default username and password, and assume that MWS is deployed at `http://localhost:8080/mws`.

This section contains these topics:

- [Javascript code samples on page 1764](#)
- [PHP code samples on page 1765](#)

- [Perl code samples on page 1769](#)
- [Python code examples on page 1771](#)
- [cURL samples on page 1771](#)

#### Related topics

- [Configuration on page 1772](#)

## Javascript code samples

When utilizing Javascript to interact with MWS, it is recommended to use libraries that provide a simple browser-independent syntax for performing REST calls. It must also be noted that it is not recommended to make calls to MWS directly from client-side Javascript, as this will contain the username and password for MWS and could potentially be retrieved and utilized by a malicious user.

Examples for specific libraries are given below.

## jQuery

The jQuery `ajax` function makes it extremely simple to perform all REST calls. There are also some shorthand methods which utilize the `ajax` method in a simple manner—such as `getJSON` and `post`—but these do not have the ability to specify authentication parameters, which is why they are not used below.

**i** In all examples given, it is assumed that the `$` variable is mapped to `jQuery`, which is the default configuration of jQuery.

### GET

```
$.ajax({
  url: "http://localhost:8080/mws/rest/jobs",
  dataType: 'json',
  username: 'admin',
  password: 'secret',
  success: new function(data) {
    $('<div class="result">').html(data);
    alert("GET was successful");
  }
});
```

## POST

```
$.ajax({
  url: "http://localhost:8080/mws/rest/jobs",
  data: {
    "commandFile":"/tmp/test.sh",
    "initialWorkingDirectory":"/tmp",
    "user":"adaptive",
    "requirements":[{"requiredNodeCountMinimum":1}]
  }
  dataType: 'json',
  username: 'admin',
  password: 'secret',
  success: new function(data) {
    $('<div class="result">').html(data);
    alert("GET was successful");
  }
});
```

## PUT

```
$.ajax({
  type: 'PUT',
  url: 'http://localhost:8080/mws/rest/jobs/Moab.1',
  data: {"holds":["user"]},
  dataType: "json",
  username: 'admin',
  password: 'secret',
  success: function(data) {
    $('<div class="result">').html(data);
    alert("PUT was successful");
  }
});
```

## DELETE

```
$.ajax({
  type: 'DELETE',
  url: 'http://localhost:8080/mws/rest/jobs/Moab.1',
  username: 'admin',
  password: 'secret',
  success: function(data) {
    $('<div class="result">').html(data);
    alert("DELETE was successful");
  }
});
```

### Related topics

- [Client code samples on page 1763](#)

## PHP code samples

All of these examples make use of the [cURL PHP Extension](#). While this library is easy to use for GET and POST requests, it is somewhat more difficult to use for PUT and DELETE requests. Each will be covered in this topic.

**i** Notice the use of the [JSON PHP Extension](#)'s `json_decode` and `json_encode` functions.

## GET

```
<?php
$baseUrl = "http://localhost:8080/mws/rest";
$resource = "/jobs";
$username = "admin";
$password = "secret";
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_TIMEOUT, 10);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
curl_setopt($ch, CURLOPT_URL, "$baseUrl$resource");

$responseBody = curl_exec($ch);
$responseInfo = curl_getinfo($ch);
curl_close($ch);

if ($responseInfo["http_code"]!=200 && $responseInfo["http_code"]!=201) {
    print_r($responseInfo);
    echo $responseBody;
} else {
    print_r(json_decode($responseBody));
}
?>
```

POST

```
<?php
$baseUrl = "http://localhost:8080/mws/rest";
$resource = "/jobs";
$username = "admin";
$password = "secret";
$requestPayload = array(
    "commandFile"=>"/tmp/test.sh",
    "initialWorkingDirectory"=>"/tmp",
    "user"=>"adaptive",
    "requirements"=>array(
        array("requiredNodeCountMinimum"=>1)
    )
);
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_TIMEOUT, 10);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: application/json"));
curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
curl_setopt($ch, CURLOPT_URL, "$baseUrl$resource");
// Setup POST request
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($requestPayload));
curl_setopt($ch, CURLOPT_POST, 1);

$responseBody = curl_exec($ch);
$responseInfo = curl_getinfo($ch);
curl_close($ch);

if ($responseInfo["http_code"]!=200 && $responseInfo["http_code"]!=201) {
    print_r($responseInfo);
    echo $responseBody;
} else {
    print_r(json_decode($responseBody));
}
?>
```

PUT

```

<?php
$baseUrl = "http://localhost:8080/mws/rest";
$resource = "/jobs/Moab.1";
$username = "admin";
$password = "secret";
$requestPayload = array(
    "holds"=>array("user")
);
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_TIMEOUT, 10);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: application/json"));
curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
curl_setopt($ch, CURLOPT_URL, "$baseUrl$resource");
// Setup PUT request
curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($requestPayload));
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "PUT");

$responseBody = curl_exec($ch);
$responseInfo = curl_getinfo($ch);
curl_close($ch);

if ($responseInfo["http_code"]!=200 && $responseInfo["http_code"]!=201) {
    print_r($responseInfo);
    echo $responseBody;
} else {
    print_r(json_decode($responseBody));
}

```

DELETE

```

<?php
$baseUrl = "http://localhost:8080/mws/rest";
$resource = "/jobs/Moab.1";
$username = "admin";
$password = "secret";
$requestPayload = array(
    "holds"=>array("user")
);
$ch = curl_init();
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
curl_setopt($ch, CURLOPT_TIMEOUT, 10);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: application/json"));
curl_setopt($ch, CURLOPT_USERPWD, "$username:$password");
curl_setopt($ch, CURLOPT_URL, "$baseUrl$resource");
// Setup DELETE request
curl_setopt($ch, CURLOPT_CUSTOMREQUEST, "DELETE");

$responseBody = curl_exec($ch);
$responseInfo = curl_getinfo($ch);
curl_close($ch);

if ($responseInfo["http_code"]!=200 && $responseInfo["http_code"]!=201) {
    print_r($responseInfo);
    echo $responseBody;
} else {
    print_r(json_decode($responseBody));
}

```

## Related topics

- [Client code samples on page 1763](#)

## Perl code samples

These examples all utilize the `LWP::UserAgent` module, which must be installed before running them.

### GET

```
#!/usr/bin/perl -w
use strict;
use warnings;
# Create a user agent object
use LWP::UserAgent;
my $ua = LWP::UserAgent->new;
$ua->agent("MyApp/0.1");

# Create a request
my $req = HTTP::Request->new(GET => 'http://localhost:8080/mws/rest/images');
$req->content_type('application/json');
$req->authorization_basic("admin", "secret");

# Pass request to the user agent and get a response back
my $res = $ua->request($req);

# Check the outcome of the response
if ($res->is_success) {
    print $res->content;
} else {
    print $res->status_line, "n";
}
```

### POST

```
#!/usr/bin/perl -w
use strict;
use warnings;
# Create a user agent object
use LWP::UserAgent;
my $ua = LWP::UserAgent->new;
$ua->agent("MyApp/0.1");

# Create a request
my $req = HTTP::Request->new(POST => 'http://localhost:8080/mws/rest/images');
$req->content_type('application/json');
$req->authorization_basic("admin", "secret");
$req->content('
{"profile":"compute","osVersion":"5","name":"centos5stateless","hypervisor":0,"architecture":"x86_64","osName":"centos","osType":"linux","type":"stateless"}');

# Pass request to the user agent and get a response back
my $res = $ua->request($req);

# Check the outcome of the response
if ($res->is_success) {
    print $res->content;
} else {
    print $res->status_line, "n";
}
```

PUT

```
#!/usr/bin/perl -w
use strict;
use warnings;
# Create a user agent object
use LWP::UserAgent;
my $ua = LWP::UserAgent->new;
$ua->agent("MyApp/0.1");

# Create a request
my $req = HTTP::Request->new(PUT => 'http://localhost:8080/mws/rest/images/centos5-
stateless');
$req->content_type('application/json');
$req->authorization_basic("admin", "secret");
$req->content('{"osVersion":"5.5"}');

# Pass request to the user agent and get a response back
my $res = $ua->request($req);

# Check the outcome of the response
if ($res->is_success) {
    print $res->content;
} else {
    print $res->status_line, "n";
}
}
```

DELETE

```
#!/usr/bin/perl -w
use strict;
use warnings;
# Create a user agent object
use LWP::UserAgent;
my $ua = LWP::UserAgent->new;
$ua->agent("MyApp/0.1");

# Create a request
my $req = HTTP::Request->new(DELETE => 'http://localhost:8080/mws/rest/images/centos5-
stateless');
$req->content_type('application/json');
$req->authorization_basic("admin", "secret");

# Pass request to the user agent and get a response back
my $res = $ua->request($req);

# Check the outcome of the response
if ($res->is_success) {
    print $res->content;
} else {
    print $res->status_line, "n";
}
}
```

## Related topics

- [Client code samples on page 1763](#)

## Python code examples

Notice the use of the `json` module to build a Python object from the return JSON data. If you want, you can also use `json.dumps` to create a JSON string from a Python object.

### Simple request (GET)

```
import urllib
import base64
import string
import json
def get(base, port, url):
    conn = urllib.HTTPConnection(base, port, timeout=60)
    conn.request('GET', url, None, { 'Authorization' : 'Basic '+string.strip
(base64.encodestring('admin:secret'))})
    return conn.getresponse().read()

data = get("localhost", 8080, "/mws/rest/jobs?format=json")
print json.loads(data)
```

### Complex request (POST)

```
import urllib
import base64
import string
import json
def post(base, port, url, payload):
    conn = urllib.HTTPConnection(base, port, timeout=60)
    conn.request('POST', url, payload, { 'Authorization' : 'Basic '+string.strip
(base64.encodestring('admin:secret')), 'Content-Type' : 'application/json' })
    r = conn.getresponse()
    return r.read()

# Note that json.dumps may also be used to create the json string from a python object
data = post("localhost", 8080, "/mws/rest/jobs", '
{"commandFile":"/tmp/test.sh","initialWorkingDirectory":"/tmp","user":"adaptive","requirements":[{"requiredNodeCountMinimum":1}]}')
print json.loads(data)
```

### Related topics

- [Client code samples on page 1763](#)

## cURL samples

Unlike the other code samples given in this section, these samples are simple commands that can be run from any server command line with the `curl` program installed in order to communicate with MWS.

### GET

```
curl -u admin:secret -X GET -H "Content-Type: application/json"
http://localhost:8080/mws/rest/jobs
```

### POST

```
curl -u admin:secret -X POST -H "Content-Type: application/json"
http://localhost:8080/mws/rest/jobs
-d '{
  "commandFile":"/tmp/test.sh", "initialWorkingDirectory":"/tmp", "user":"adaptive", "requirements":[{"requiredNodeCountMinimum":1}]}'
```

### PUT

```
curl -u admin:secret -X PUT -H "Content-Type: application/json"
http://localhost:8080/mws/rest/jobs
-d '{"holds":["user"]}'
```

### DELETE

```
curl -u admin:secret -X DELETE -H "Content-Type: application/json"
http://localhost:8080/mws/rest/jobs
```

### Related topics

- [Client code samples on page 1763](#)

## Configuration

These properties can be modified by setting the appropriate values in the `mws-config.groovy` file. This file is located in `MWS_HOME/etc/` or `/opt/mws/etc/` by default as explained in [Configuring Moab Web Services on page 1395](#).

**i** The configuration file is read not only on startup, but also each time it is changed. Several properties, including those for Moab Workload Manager (`moab`), Moab Accounting Manager (`mam`), Mongo (`grails.mongo`), and authentication (`auth`) are processed after each change and can affect the runtime behavior of MWS.

Configuration files can also be placed in the `MWS_HOME/etc/mws.d` directory. Any configuration files here get merged with `MWS_HOME/etc/mws-config.groovy`. In case of conflict, the configuration in `MWS_HOME/etc/mws.d` takes precedence.

### Configuration reference

For all possible values that can be set, please see the Grails reference guide. For project specific settings (usually the only ones you'll need to change), you may set the following properties:

Property	Type	Default	Description
<code>auth.defaultUser.password</code>	String	changeme!	Unencoded password of the default admin user.

Property	Type	Default	Description
<code>auth.defaultUser.username</code>	String	moab-admin	Username of the default admin user (only created if no other users exist).
<code>grails.mongo.host</code>	String	127.0.0.1	The MongoDB host to use (Note that MongoDB runs on 127.0.0.1 and <i>not</i> localhost by default).
<code>dataSource.insight.password</code>	String	changeme!	The password for the username used to log in to the Insight database.
<code>dataSource.insight.url</code>	String	jdbc:postgresql://127.0.0.1:5432/n insight	The <a href="#">JDBC URL</a> for the Insight database. For more information, see <a href="#">Configuring Moab Web Services on page 1395</a> .
<code>dataSource.insight.username</code>	String	mws	The username used to log into the Insight database.
<code>grails.mongo.port</code>	Integer	27017	The MongoDB port to use.
<code>grails.mongo.replicaSet</code>	List of Strings	n/a	The MongoDB replica set servers to use (for example, ["moab1:27017", "moab2:27017"]); <b>note</b> that <code>grails.mongo.host</code> <i>must</i> be set to null to use this option.
<code>grails.mongo.databaseName</code>	String	mws	The MongoDB database name to use.

Property	Type	Default	Description
<b>grails.mongo.username</b>	String	-	(Optional) The user-name to use when connecting to MongoDB.
<b>grails.mongo.password</b>	String	-	(Optional) The password to use when connecting to MongoDB.
<b>grails.mongo.options.connectionsPerHost</b>	Integer	50	The number of connections allowed per host.
<b>grails.- mongo.- options.threadsAllowedToBlockForConnecti</b>	Integer	5	The number of threads per connection allowed to wait for an available connection.
<b>grails.mongo.options.autoConnectRetry</b>	Boolean	true	Controls whether the system retries automatically on connection errors.
<b>grails.mime.use.accept.header</b>	Boolean	false	When enabled, uses the HTTP Content-Accept header to determine the content type used for return data (JSON only for now).
<b>grails.plu- gins.springsecurity.basic.realmName</b>	String	Moab Web Services	The HTTP realm used when using basic auth.
<b>grails.plugins.springsecurity.active</b>	Boolean	true	Enables or disables security for MWS as a whole, including all providers.
<b>grails.plugins.springsecurity.useBasicAuth</b>	Boolean	true	Enables or disables basic auth with a simple user-name/password.

Property	Type	Default	Description
<b>grails.plugins.springsecurity.oauthProvider.active</b>	Boolean	true	Enables or disables the OAuth2 provider.
<b>ldap.baseDNs</b>	List of Strings	-	A list of distinguished names that are the root entries for LDAP searches.
<b>ldap.bindUser</b>	String	-	The distinguished name of the LDAP bind user.
<b>ldap.directory.type</b>	String	-	The type of LDAP directory (for example, "Microsoft Active Directory"). See <a href="#">Configuring Moab Web Services on page 1395</a> for valid values..
<b>ldap.password</b>	String	-	The password of the LDAP bind user
<b>ldap.port</b>	Integer	-	LDAP server's port
<b>ldap.security.server.certificate</b>	String	-	The filename of the LDAP server's PEM encoded X.509 certificate. See <a href="#">Setting up MWS security on page 1410</a> for more information.
<b>ldap.security.type</b>	String	-	How the connection between MWS and LDAP is secured. See <a href="#">Setting up MWS security on page 1410</a> for more information.
<b>ldap.server</b>	String	-	LDAP server hostname or IP address

Property	Type	Default	Description
<b>mam.server</b>	String	localhost	Moab Accounting Manager server hostname or IP address
<b>mam.port</b>	Integer	7112	Moab Accounting Manager server's port
<b>mam.secretKey</b>	String	mamsecret	Secret key used to communicate with Moab Accounting Manager
<b>moab.databaseName</b>	String	moab	The name of the MongoDB database to use to retrieve current Moab data; this should match the database setting in Moab.
<b>moab.messageQueue.port</b>	Integer	5563	The port on which Moab publishes ZeroMQ messages.
<b>moab.messageQueue.secretKey</b>	String	-	Used to encrypt and decrypt messages on the message queue using AES. Must be a <a href="#">Base64</a> -encoded 128-bit (16-byte) key. For example: "1r6RvfqJa6voezy5wAx0hw=="
<b>moab.port</b>	Integer	42559	Moab server's port
<b>moab.secretKey</b>	String	moabsecret	Secret key used to communicate with Moab. See Moab Configuration.
<b>moab.server</b>	String	localhost	Moab server hostname or IP address

Property	Type	Default	Description
<b>mws.cache.duration.default</b>	Integer	60	The default number of seconds to use for caching objects from Moab. This is only supported in certain objects such as policies.
<b>mws.cache.duration.policy</b>	Integer	180	The number of seconds that the cache for policies is valid. If set to null, the default is used.
<b>mws.certificates.location</b>	String	etc/ssl.crt	The directory (relative or absolute) where plugin certificates are stored. See the <a href="#">Managing SSL connections on page 1714</a> .
<b>mws.events.expireAfterSeconds</b>	Integer	2592000	Events older than this many seconds (30 days by default) will be deleted from the database. Effective only with MongoDB 2.2 or later.
<b>mws.health.check.period</b>	Integer	30	The number of seconds in between health checks. Used in creating notification conditions if problems exist in configuration or connections. For more information, see <a href="#">Notification conditions on page 1580</a> .

Property	Type	Default	Description
<b>mws.hooks.location</b>	String	hooks	The directory (relative or absolute) where Hooks are stored. See <a href="#">Pre and post-processing hooks on page 1434</a> for more information.
<b>mws.plugins.location</b>	String	plugins	The directory (relative or absolute) where Plugins are stored. See <a href="#">About Moab Web Services plugins on page 1672</a> for more information.
<b>mws.messageQueue.port</b>	Integer	5570	The port on which MWS publishes ZeroMQ messages.
<b>mws.messageQueue.address</b>	String	-	The IP address on which MWS publishes ZeroMQ messages.
<b>mws.services.hooks.syncInterval</b>	Integer	30	The number of seconds between each time MWS checks for service phase transition hooks that completed or timed out.
<b>mws.services.phases.syncInterval</b>	Integer	14400	The number of seconds between each time MWS checks with Moab Workload Manager to verify that the service phases are correctly synchronized.
<b>mws.suite</b>	String	CLOUD	The suite or context that MWS is running in (see <a href="#">Suite</a> for valid values)

Property	Type	Default	Description
<b>pam.configuration.service</b>	String	-	The name of the PAM configuration file located in <code>/etc/pam.d</code> . This parameter and specification tells MWS which PAM configuration file you want to use. For more information, see <a href="#">PAM (pluggable authentication module) configuration using mws-config.groovy</a> on page 1405.
<b>plugins.pluginType</b>	String	-	Default configuration value for the plugin <code>pluginType</code> field (see <a href="#">Setting default plugin configuration</a> on page 1747).
<b>plugins.autoStart</b>	Boolean	true	Default configuration value for the plugin <code>autoStart</code> field (see <a href="#">Setting default plugin configuration</a> on page 1747).
<b>plugins.pollInterval</b>	Integer	30	Default configuration value for the plugin <code>pollInterval</code> field (see <a href="#">Setting default plugin configuration</a> on page 1747).
<b>plugins.config</b>	Map	-	Default configuration value for the plugin <code>config</code> field (see <a href="#">Setting default plugin configuration</a> on page 1747).

Property	Type	Default	Description
<b>plugins.loadInitialPlugins</b>	Boolean	true	If true, loads the initial plugins defined for uploaded or built-in plugin types (see <a href="#">Plugin projects and metadata on page 1721</a> ).
<b>plugins.stateConsolidationPolicy</b>	NodeStatePolicy	null	If "optimistic", treats state data optimistically. If "pessimistic", treats state state pessimistically. May be null. See <a href="#">Data consolidation on page 1677</a> for more information.
<b>plugins.defaultHypervisorType</b>	String	ESX	This is reported to Moab when a node report references a hypervisor image that does not have the hypervisorType or extensions.xcat.hvType fields set. See <a href="#">Fields: Images on page 1903</a> .

### [Logging reference](#)

The following loggers are available to use for debugging purposes:

Logger	Default	Description
<b>grails.app</b>	debug	Most classes in the main MWS application.
<b>grails.app.bootstrap.BootStrap</b>	debug	Handles startup and initialization of MWS.

Logger	Default	Description
<b>com.ace.mws</b>	debug	The base logger for MWS specific functionality not included in other loggers (this comprises very few classes).
<b>grails.app.services.com.ace.mws.plugins.PluginUtilityService</b>	debug	Class for initializing and helper methods of plugins.
<b>com.ace.mws.hooks.HookUtils</b>	debug	Helper class for loading hooks during startup process.
<b>plugins</b>	debug	All MWS plugins (see <a href="#">About Moab Web Services plugins on page 1672</a> ).
<b>com.ace.mws.plugins</b>	debug	MWS plugin helper class, used to create and initialize plugins.
<b>com.ace.mws.gapi</b>	warn	Base logger for all Moab connections, requests, and responses.
<b>com.ace.mws.gapi.Connection</b>	info	Logger which controls all requests and responses from Moab.
<b>com.ace.mws.gapi.parsers</b>	info	Loggers for parsers of Moab's data.
<b>com.ace.mws.gapi.serializers</b>	info	Loggers for all serialization from MWS to Moab Wire Protocol.
<b>grails.app.service.grails.plugins.reloadconfig</b>	info	Handles dynamic reloading of configuration files.
<b>net.sf.json</b>	error	JSON and XML processing library.
<b>org.springframework.security</b>	info	Authentication/authorization logger.

Logger	Default	Description
<code>org.codehaus.groovy.grails.web.servlet</code>	error	Loggers for request handlers.
<code>org.codehaus.groovy.grails.web.mapping</code>	error	URL mapping.
<code>org.codehaus.groovy.grails.web.mapping.filter</code>	error	URL mapping.
<code>org.codehaus.groovy.grails.plugins</code>	error	All grails plugins (MWS internal).
<code>org.codehaus.groovy.grails.commons</code>	error	Core application and class-loading.

### Related topics

- [Configuring Moab Web Services on page 1395](#)

## Resource reference

### Resources reference

This section contains the type and description of all possible fields in each MWS resource object. Because of significant changes in the API introduced between releases, MWS possesses a versioned API. Each resource contains drop-down sections for each API version.

This section contains these topics:

- [Fields: Access Control Lists \(ACLs\) on page 1783](#)
- [Fields: Accounts on page 1796](#)
- [Fields: Allocations on page 1800](#)
- [Fields: Charge Rates on page 1806](#)
- [Fields: Credentials on page 1893](#)
- [Fields: Events on page 1894](#)
- [Fields: Fund Balances on page 1809](#)
- [Fields: Fund Statements on page 1835](#)
- [Fields: Fund Statement Summary on page 1819](#)
- [Fields: Funds on page 1850](#)
- [Fields: Images on page 1903](#)
- [Fields: Job Arrays on page 1915](#)

- [Fields: Job Templates](#) on page 2107
- [Fields: Jobs](#) on page 2006
- [Fields: Liens](#) on page 1862
- [Fields: Metric Types](#) on page 2149
- [Fields: Nodes](#) on page 2150
- [Fields: Notification Conditions](#) on page 2191
- [Fields: Notifications](#) on page 2195
- [Fields: Organizations](#) on page 1868
- [Fields: User's Permissions](#) on page 2428
- [Fields: Plugins](#) on page 2197
- [Fields: Plugin Types](#) on page 2204
- [Fields: Policies](#) on page 2210
- [Fields: Principals](#) on page 2249
- [Fields: Quotes](#) on page 1871
- [Fields: Report Datapoints](#) on page 2259
- [Fields: Report Samples](#) on page 2341
- [Fields: Reports](#) on page 2261
- [Fields: Reservations](#) on page 2270
- [Fields: Resource Types](#) on page 2330
- [Fields: Roles](#) on page 2331
- [Fields: Standing Reservations](#) on page 2343
- [Fields: Transactions](#) on page 1878
- [Fields: Usage Records](#) on page 1884
- [Fields: Users](#) on page 1890

#### Related topics

- [Resources introduction](#) on page 1446
- [Global URL parameters](#) on page 1425

## Fields: Access Control Lists (ACLs)



See the associated [Access control lists \(ACLs\)](#) on page 1448 resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	acl-rules	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	acl-rules.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

AclRule

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#">AclAffinity.POSITIVE</a> .
<b>comparator</b>	<a href="#">ComparisonOperator</a>	Yes	The type of comparison to make against the ACL object.  Defaults to <a href="#">ComparisonOperator.EQUAL</a> .
<b>type</b>	<a href="#">AclType</a>	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	Yes	The name of the object that is being granted (or denied) access.

AclAffinity

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported

Value	Description
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

## API version 2

AclRule

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#">AclAffinity.POSITIVE</a> .
<b>comparator</b>	<a href="#">ComparisonOperator</a>	Yes	The type of comparison to make against the ACL object.  Defaults to <a href="#">ComparisonOperator.EQUAL</a> .
<b>type</b>	<a href="#">AclType</a>	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	Yes	The name of the object that is being granted (or denied) access.

AclAffinity

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported

Value	Description
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

## API version 1

AclRule

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#">AclAffinity.POSITIVE</a> .
<b>comparator</b>	<a href="#">ComparisonOperator</a>	Yes	The type of comparison to make against the ACL object.  Defaults to <a href="#">ComparisonOperator.EQUAL</a> .
<b>type</b>	<a href="#">AclType</a>	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	Yes	The name of the object that is being granted (or denied) access.

AclAffinity

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported

Value	Description
QUEUE	Not supported
RACK	Not supported
SCHED	Not supported
SYSTEM	Not supported
TASK	Not supported
VC	Not supported
XFACTOR	Not supported

Related topics

- [Access control lists \(ACLs\) on page 1448](#)

## Accounting

**Fields: Accounts**

 See the associated [Accounting Accounts on page 1451](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/accounts	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.accounts.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Account

Users may be designated as members of an account and may be allowed to share its allocations. The user members may be designated as active or inactive, and as an account admin or not an account admin. Default account properties include the description, the organization it is part of, and whether or not it is active. An account's user membership can also be adjusted. By default, a standard user may only query accounts they belong to.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique account identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this account is active or not
<b>creationTime</b>	<b>Date</b>	The time this account was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this account is deleted or not
<b>description</b>	<b>String</b>	The account description
<b>modificationTime</b>	<b>Date</b>	The time this account was last modified
<b>organization</b>	<b>String</b>	The organization to which the account belongs
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>users</b>	<a href="#">Set&lt;AccountUser&gt;</a>	The users associated with this account

AccountUser

An account user is a person authorized to use an account.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>admin</b>	<b>Boolean</b>	A boolean indicating wheter this user is an admin or not

API version 2

Account

Users may be designated as members of an account and may be allowed to share its allocations. The user members may be designated as active or inactive, and as an account admin or not an account admin. Default account properties include the description, the organization it is part of, and whether or not it is active. An account's user membership can also be adjusted. By default, a standard user may only query accounts they belong to.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique account identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this account is active or not
<b>creationTime</b>	<b>Date</b>	The time this account was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this account is deleted or not
<b>description</b>	<b>String</b>	The account description
<b>modificationTime</b>	<b>Date</b>	The time this account was last modified
<b>organization</b>	<b>String</b>	The organization to which the account belongs
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>users</b>	<a href="#">Set&lt;AccountUser&gt;</a>	The users associated with this account

AccountUser

An account user is a person authorized to use an account.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>admin</b>	<b>Boolean</b>	A boolean indicating wheter this user is an admin or not

## API version 1

Account

Users may be designated as members of an account and may be allowed to share its allocations. The user members may be designated as active or inactive, and as an account admin or not an account admin. Default account properties include the description, the organization it is part of, and whether or not it is active. An account's user membership can also be adjusted. By default, a standard user may only query accounts they belong to.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique account identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this account is active or not
<b>creationTime</b>	<b>Date</b>	The time this account was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this account is deleted or not
<b>description</b>	<b>String</b>	The account description
<b>modificationTime</b>	<b>Date</b>	The time this account was last modified
<b>organization</b>	<b>String</b>	The organization to which the account belongs
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>users</b>	<a href="#">Set&lt;AccountUser&gt;</a>	The users associated with this account

AccountUser

An account user is a person authorized to use an account.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>admin</b>	<b>Boolean</b>	A boolean indicating wheter this user is an admin or not

Related topics

- [Accounting Accounts on page 1451](#)

**Fields: Allocations**

 See the associated [Accounting Allocations on page 1455](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/allocations	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.allocations.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified

Field Name	Type	Description
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## API version 2

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified

Field Name	Type	Description
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## API version 1

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified

Field Name	Type	Description
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### Related topics

- [Accounting Allocations on page 1455](#)

#### Fields: Charge Rates

 See the associated [Accounting Charge rates on page 1459](#) resource section for more information on how to use this resource and supported operations.

#### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/charge-rates	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.charge-rates.-groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

ChargeRate

Charge rates establish how much to charge for usage. A charge rate consists of its name, an optional value and the amount. Both name and value are primary keys and a charge rate is uniquely defined by both its name and its value. A charge rate value that is null designates the default charge rate.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>creationTime</b>	<b>Date</b>	The date this charge rate was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this charge rate is deleted or not
<b>description</b>	<b>String</b>	The charge rate description
<b>modificationTime</b>	<b>Date</b>	The date this charge rate was last modified
<b>name</b>	<b>String</b>	The charge rate name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>value</b>	<b>String</b>	The charge rate value. This will be null for default charge rates.

## API version 2

ChargeRate

Charge rates establish how much to charge for usage. A charge rate consists of its name, an optional value and the amount. Both name and value are primary keys and a charge rate is uniquely defined by both its name and its value. A charge rate value that is null designates the default charge rate.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>creationTime</b>	<b>Date</b>	The date this charge rate was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this charge rate is deleted or not
<b>description</b>	<b>String</b>	The charge rate description
<b>modificationTime</b>	<b>Date</b>	The date this charge rate was last modified
<b>name</b>	<b>String</b>	The charge rate name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>value</b>	<b>String</b>	The charge rate value. This will be null for default charge rates.

## API version 1

ChargeRate

Charge rates establish how much to charge for usage. A charge rate consists of its name, an optional value and the amount. Both name and value are primary keys and a charge rate is uniquely defined by both its name and its value. A charge rate value that is null designates the default charge rate.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>creationTime</b>	<b>Date</b>	The date this charge rate was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this charge rate is deleted or not
<b>description</b>	<b>String</b>	The charge rate description
<b>modificationTime</b>	<b>Date</b>	The date this charge rate was last modified
<b>name</b>	<b>String</b>	The charge rate name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>value</b>	<b>String</b>	The charge rate value. This will be null for default charge rates.

## Related topics

- [Accounting Charge rates on page 1459](#)

**Fields: Fund Balances**

 See the associated [Accounting Funds on page 1463](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/funds/balances	<a href="#"><u>Permissions on page 1593</u></a>
<b>Hooks filename</b>	accounting.funds.balances.groovy	<a href="#"><u>Pre and post-processing hooks on page 1434</u></a>
<b>Distinct query-supported</b>	No	<a href="#"><u>Distinct on page 1526</u></a>

## API version 3

FundBalance

Represents a report of fund balance.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	The total adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	Allocations associated with this fund
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>available</b>	<b>BigDecimal</b>	The total amount available for charging. <a href="#">amount</a> - <a href="#">reserved</a> + <a href="#">creditLimit</a>
<b>balance</b>	<b>BigDecimal</b>	The allocation total not blocked by liens. <a href="#">amount</a> - <a href="#">reserved</a>
<b>capacity</b>	<b>BigDecimal</b>	The total amount allocated via deposits and credit limits. <a href="#">allocated</a> + <a href="#">creditLimit</a>
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>percentRemaining</b>	<b>Double</b>	The percentage of allocation remaining. <a href="#">amount</a> * 100 / <a href="#">allocated</a>

Field Name	Type	Description
<b>percentUsed</b>	<b>Double</b>	The percentage of allocated used. $\text{used} * 100 / \text{allocated}$
<b>reserved</b>	<b>BigDecimal</b>	The sum of active lien amounts against this fund
<b>used</b>	<b>BigDecimal</b>	The total amount used this allocation cycle. $\text{allocated} - \text{amount}$

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

## API version 2

FundBalance

Represents a report of fund balance.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	The total adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	Allocations associated with this fund
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>available</b>	<b>BigDecimal</b>	The total amount available for charging. <a href="#">amount</a> - <a href="#">reserved</a> + <a href="#">creditLimit</a>
<b>balance</b>	<b>BigDecimal</b>	The allocation total not blocked by liens. <a href="#">amount</a> - <a href="#">reserved</a>
<b>capacity</b>	<b>BigDecimal</b>	The total amount allocated via deposits and credit limits. <a href="#">allocated</a> + <a href="#">creditLimit</a>
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>percentRemaining</b>	<b>Double</b>	The percentage of allocation remaining. <a href="#">amount</a> * 100 / <a href="#">allocated</a>

Field Name	Type	Description
<b>percentUsed</b>	<b>Double</b>	The percentage of allocated used. $\frac{\text{used} * 100}{\text{allocated}}$
<b>reserved</b>	<b>BigDecimal</b>	The sum of active lien amounts against this fund
<b>used</b>	<b>BigDecimal</b>	The total amount used this allocation cycle. $\text{allocated} - \text{amount}$

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

## API version 1

FundBalance

Represents a report of fund balance.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	The total adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	Allocations associated with this fund
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>available</b>	<b>BigDecimal</b>	The total amount available for charging. <a href="#">amount</a> - <a href="#">reserved</a> + <a href="#">creditLimit</a>
<b>balance</b>	<b>BigDecimal</b>	The allocation total not blocked by liens. <a href="#">amount</a> - <a href="#">reserved</a>
<b>capacity</b>	<b>BigDecimal</b>	The total amount allocated via deposits and credit limits. <a href="#">allocated</a> + <a href="#">creditLimit</a>
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>percentRemaining</b>	<b>Double</b>	The percentage of allocation remaining. <a href="#">amount</a> * 100 / <a href="#">allocated</a>

Field Name	Type	Description
<b>percentUsed</b>	<b>Double</b>	The percentage of allocated used. $\frac{\text{used} * 100}{\text{allocated}}$
<b>reserved</b>	<b>BigDecimal</b>	The sum of active lien amounts against this fund
<b>used</b>	<b>BigDecimal</b>	The total amount used this allocation cycle. $\text{allocated} - \text{amount}$

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

### Related topics

- [Accounting Funds on page 1463](#)

### Fields: Fund Statement Summary

**i** See the associated [Accounting Funds on page 1463](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/funds/reports/statement	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.funds.reports.statement.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

FundStatementSummary

An fund statement summary is related to and quite similar to the [FundStatement](#) report, but differs in the [transactions](#) field by using the [FundTransactionSummary](#).

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransactionSummary&gt;</a>	Summaries of the specific transactions which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such

as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into a fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with a fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#"><u>Set&lt;Allocation&gt;</u></a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#"><u>Set&lt;FundConstraint&gt;</u></a>	Constraints on fund usage.

Field Name	Type	Description
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation

Field Name	Type	Description
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

FundTransactionSummary

Represents a Moab Accounting Manager transaction summary, which is a consolidated view of multiple transactions. The transactions are grouped by object and action, and a total count is given for the summary.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>count</b>	<b>Long</b>	The number of transactions in this grouping of object and action
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>object</b>	<b>String</b>	Object's name associated with the transaction

API version 2

FundStatementSummary

An fund statement summary is related to and quite similar to the [FundStatement](#) report, but differs in the [transactions](#) field by using the [FundTransactionSummary](#).

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransactionSummary&gt;</a>	Summaries of the specific transactions which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such

as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.

Field Name	Type	Description
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation

Field Name	Type	Description
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

FundTransactionSummary

Represents a Moab Accounting Manager transaction summary, which is a consolidated view of multiple transactions. The transactions are grouped by [object](#) and [action](#), and a total [count](#) is given for the summary.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>count</b>	<b>Long</b>	The number of transactions in this grouping of object and action
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>object</b>	<b>String</b>	Object's name associated with the transaction

## API version 1

FundStatementSummary

An fund statement summary is related to and quite similar to the [FundStatement](#) report, but differs in the [transactions](#) field by using the [FundTransactionSummary](#).

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransactionSummary&gt;</a>	Summaries of the specific transactions which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such

as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into a fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with a fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.

Field Name	Type	Description
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation

Field Name	Type	Description
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

FundTransactionSummary

Represents a Moab Accounting Manager transaction summary, which is a consolidated view of multiple transactions. The transactions are grouped by [object](#) and [action](#), and a total [count](#) is given for the summary.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>count</b>	<b>Long</b>	The number of transactions in this grouping of object and action
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>object</b>	<b>String</b>	Object's name associated with the transaction

Related topics

- [Accounting Funds on page 1463](#)

**Fields: Fund Statements**

 See the associated [Accounting Funds on page 1463](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/funds/reports/statement	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.funds.reports.statement.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

API version 3

FundStatement

An fund statement is a report generated from Moab Accounting Manager fund, allocation, and transaction data. It contains fields detailing the specific time period covered, the starting and ending balances, the total of the transactions, and fund and transaction details.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransaction&gt;</a>	Details of each specific transaction which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints

may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount

Field Name	Type	Description
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created

Field Name	Type	Description
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

#### FundTransaction

Represents a Moab Accounting Manager transaction.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>account</b>	<b>String</b>	The account associated with the transaction. For a credit this will likely be zero
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>instance</b>	<b>String</b>	Instance name
<b>machine</b>	<b>String</b>	The machine associated with the transaction. For a credit this will likely be zero. This field is not available in the Cloud context.
<b>object</b>	<b>String</b>	Object's name associated with the transaction
<b>time</b>	<b>Date</b>	The date at which the transaction occurred
<b>user</b>	<b>String</b>	The user associated with the transaction. For a credit this will likely be zero

## API version 2

FundStatement

An fund statement is a report generated from Moab Accounting Manager fund, allocation, and transaction data. It contains fields detailing the specific time period covered, the starting and ending balances, the total of the transactions, and fund and transaction details.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransaction&gt;</a>	Details of each specific transaction which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints

may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#"><u>Set&lt;Allocation&gt;</u></a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#"><u>Set&lt;FundConstraint&gt;</u></a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount

Field Name	Type	Description
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created

Field Name	Type	Description
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

#### FundTransaction

Represents a Moab Accounting Manager transaction.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>account</b>	<b>String</b>	The account associated with the transaction. For a credit this will likely be zero
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>instance</b>	<b>String</b>	Instance name
<b>machine</b>	<b>String</b>	The machine associated with the transaction. For a credit this will likely be zero. This field is not available in the Cloud context.
<b>object</b>	<b>String</b>	Object's name associated with the transaction
<b>time</b>	<b>Date</b>	The date at which the transaction occurred
<b>user</b>	<b>String</b>	The user associated with the transaction. For a credit this will likely be zero

API version 1

FundStatement

An fund statement is a report generated from Moab Accounting Manager fund, allocation, and transaction data. It contains fields detailing the specific time period covered, the starting and ending balances, the total of the transactions, and fund and transaction details.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>endBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">endTime</a> of the statement
<b>endTime</b>	<b>Date</b>	The ending time that the statement covers
<b>funds</b>	<a href="#">Set&lt;Fund&gt;</a>	The funds that this statement covers. Only a sub-set of the full fund fields are available from this property. This includes id, name, priority, description, and creationTime.
<b>generationTime</b>	<b>Date</b>	The date that the statement report was generated
<b>startBalance</b>	<b>BigDecimal</b>	The balance of the funds at the <a href="#">startTime</a> of the statement
<b>startTime</b>	<b>Date</b>	The starting time that the statement covers
<b>totalCredits</b>	<b>BigDecimal</b>	The total number of credits that occurred during the time period that the statement covers
<b>totalDebits</b>	<b>BigDecimal</b>	The total number of debits that occurred during the time period that the statement covers
<b>transactions</b>	<a href="#">Set&lt;FundTransaction&gt;</a>	Details of each specific transaction which occurred during the time period that this statement covers.

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints

may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#">Set&lt;Allocation&gt;</a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount

Field Name	Type	Description
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created

Field Name	Type	Description
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

#### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

#### FundTransaction

Represents a Moab Accounting Manager transaction.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>account</b>	<b>String</b>	The account associated with the transaction. For a credit this will likely be zero
<b>action</b>	<b>String</b>	Action name for the transaction
<b>amount</b>	<b>BigDecimal</b>	Amount of the transaction. A positive or amount signifies a credit, while a negative or zero amount signifies a debit.
<b>instance</b>	<b>String</b>	Instance name
<b>machine</b>	<b>String</b>	The machine associated with the transaction. For a credit this will likely be zero. This field is not available in the Cloud context.
<b>object</b>	<b>String</b>	Object's name associated with the transaction
<b>time</b>	<b>Date</b>	The date at which the transaction occurred
<b>user</b>	<b>String</b>	The user associated with the transaction. For a credit this will likely be zero

Related topics

- [Accounting Funds on page 1463](#)

Fields: Funds

 See the associated [Accounting Funds on page 1463](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/funds	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.funds.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#"><u>Set&lt;Allocation&gt;</u></a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not

Field Name	Type	Description
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

## API version 2

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#"><u>Set&lt;Allocation&gt;</u></a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#"><u>Set&lt;FundConstraint&gt;</u></a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not

Field Name	Type	Description
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

## API version 1

Fund

A fund is a container for a time-bounded reference currency called credits for which the usage is restricted by constraints that define how the credits must be used. Much like with a bank, an fund is a repository for these resource or service credits which are added through deposits and debited through withdrawals and charges. Each fund has a set of constraints designating which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, Feature, etc.). Fund constraints may also be negated with an exclamation point leading the constraint value.

When credits are deposited into an fund, they are associated with a time period within which they are valid. These time-bounded pools of credits are known as allocations. (An allocation is a pool of billable units associated with an fund for use during a particular time period.) By using multiple allocations that expire in regular intervals it is possible to implement a use-it-or-lose-it policy and establish an allocation cycle.

Funds may be nested. Hierarchically nested funds may be useful for the delegation of management roles and responsibilities. Deposit shares may be established that assist to automate a trickle-down effect for funds deposited at higher level funds. Additionally, an optional overflow feature allows charges against lower level funds to trickle up the hierarchy.

Funds may have an arbitrary name which is not necessarily unique for the fund. Funds may also have a priority which will influence the order of fund selection when charging.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique fund identifier
<b>allocated</b>	<b>BigDecimal</b>	Total Adjusted allocations. This value is affected positively by deposits, activations and destination transfers and affected negatively by withdrawals, deactivations and source transfers that have occurred since the last reset.
<b>allocations</b>	<a href="#"><u>Set&lt;Allocation&gt;</u></a>	The allocations associated with this fund.
<b>amount</b>	<b>BigDecimal</b>	The sum of active allocation amounts within this fund. It does not take into fund current liens.
<b>creationTime</b>	<b>Date</b>	Date this fund was created
<b>creditLimit</b>	<b>BigDecimal</b>	The sum of active credit limits within this fund
<b>defaultDeposit</b>	<b>String</b>	The default deposit amount

Field Name	Type	Description
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this fund is deleted or not
<b>description</b>	<b>String</b>	The fund description
<b>fundConstraints</b>	<a href="#">Set&lt;FundConstraint&gt;</a>	Constraints on fund usage.
<b>initialDeposit</b>	<b>BigDecimal</b>	The initial deposit amount
<b>modificationTime</b>	<b>Date</b>	The date this fund was last modified
<b>name</b>	<b>String</b>	The name of this fund
<b>priority</b>	<b>Integer</b>	The fund priority
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### Allocation

An allocation is a time-bounded pool of resource or service credits associated with an fund. An fund may have multiple allocations, each for use during a different time period.

An allocation has a start time and an end time that defines the time period during which the allocation may be used. By default an allocation is created with an unbounded time period (-infinity to infinity). An active flag is automatically updated to true if the fund is within its valid timeframe or false if it is not. An allocation may also have a credit limit representing the amount by which it can go negative. Thus, by having a positive balance in the Amount field, the fund is like a debit fund, implementing a pay-first use-later model. By establishing a credit limit instead of depositing an initial balance, the fund will be like a credit fund, implementing a use-first pay-later model. These strategies can be combined by depositing some amount of funds coupled with a credit limit, implementing a form of overdraft protection where the funds will be used down to the negative of the credit limit.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this allocation
<b>active</b>	<b>Boolean</b>	Indicates whether this allocation is active or not

Field Name	Type	Description
<b>allocated</b>	<b>BigDecimal</b>	Adjusted allocation. This value stores the effective allocated amount based on the initial deposit and subsequent allocation adjustments via deposits, withdrawals or transfers.
<b>amount</b>	<b>BigDecimal</b>	The amount of this allocation
<b>creationTime</b>	<b>Date</b>	The date this allocation was created
<b>creditLimit</b>	<b>BigDecimal</b>	Determines how far in the negative this allocation is permitted to be used (enforced in quotes and liens)
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this allocation is deleted or not
<b>description</b>	<b>String</b>	The description of this allocation
<b>endTime</b>	<b>Date</b>	The date this allocation becomes inactive
<b>fund</b>	<b>String</b>	The fund Id associated with this allocation
<b>modificationTime</b>	<b>Date</b>	The date this allocation was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The date this allocation becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

### FundConstraint

Constraints designate which entities (such as Users, Accounts, Machines, Classes, Organizations, etc.) may access the encapsulated credits in a fund or for which aspects of usage the funds are intended (QualityOfService, GeographicalArea, etc.).

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier of this constraint.
<b>fund</b>	<b>String</b>	The fund ID that this constraint is associated with.

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the constraint.
<b>value</b>	<b>String</b>	The value of the constraint. The constraint may be negated by the used of an exclamation point leading the value.

Related topics

- [Accounting Funds on page 1463](#)

**Fields: Liens**

 See the associated [Accounting Liens on page 1473](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/liens	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.liens.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Lien

A lien is a reservation or hold placed against an allocation. Before usage of a resource or service begins, a lien is placed against one or more allocations within the requesting user's applicable funds. Subsequent usage requests will also post liens while the available balance (active allocations minus liens) allows. When the usage ends, the lien is removed and the actual charge is made to the allocation(s). This procedure ensures that usage will only be permitted so long as the requestors have sufficient funds.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique lien identifier
<b>allocations</b>	<a href="#">Set&lt;LienAllocation&gt;</a>	The allocation amounts reserved with this lien.
<b>creationTime</b>	<b>Date</b>	The date this lien was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this lien is deleted or not
<b>description</b>	<b>String</b>	The lien description
<b>duration</b>	<b>Long</b>	The expected duration of the reserved usage in seconds
<b>endTime</b>	<b>Date</b>	The time the lien becomes inactive
<b>instance</b>	<b>String</b>	The lien is against the specified instance (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this lien was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the lien becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The id of the usage record associated with the lien and containing the usage properties

LienAllocation

Amounts of the allocations that the lien has holds against

Field Name	Type	Description
<b>id</b>	<b>String</b>	The child allocation id
<b>amount</b>	<b>Long</b>	The amount reserved against the allocation by this lien
<b>fund</b>	<b>Long</b>	The fund that the allocation is in
<b>lien</b>	<b>String</b>	The parent lien id

## API version 2

Lien

A lien is a reservation or hold placed against an allocation. Before usage of a resource or service begins, a lien is placed against one or more allocations within the requesting user's applicable funds. Subsequent usage requests will also post liens while the available balance (active allocations minus liens) allows. When the usage ends, the lien is removed and the actual charge is made to the allocation(s). This procedure ensures that usage will only be permitted so long as the requestors have sufficient funds.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique lien identifier
<b>allocations</b>	<a href="#">Set&lt;LienAllocation&gt;</a>	The allocation amounts reserved with this lien.
<b>creationTime</b>	<b>Date</b>	The date this lien was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this lien is deleted or not
<b>description</b>	<b>String</b>	The lien description
<b>duration</b>	<b>Long</b>	The expected duration of the reserved usage in seconds
<b>endTime</b>	<b>Date</b>	The time the lien becomes inactive
<b>instance</b>	<b>String</b>	The lien is against the specified instance (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this lien was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the lien becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The id of the usage record associated with the lien and containing the usage properties

LienAllocation

Amounts of the allocations that the lien has holds against

Field Name	Type	Description
<b>id</b>	<b>String</b>	The child allocation id
<b>amount</b>	<b>Long</b>	The amount reserved against the allocation by this lien
<b>fund</b>	<b>Long</b>	The fund that the allocation is in
<b>lien</b>	<b>String</b>	The parent lien id

## API version 1

Lien

A lien is a reservation or hold placed against an allocation. Before usage of a resource or service begins, a lien is placed against one or more allocations within the requesting user's applicable funds. Subsequent usage requests will also post liens while the available balance (active allocations minus liens) allows. When the usage ends, the lien is removed and the actual charge is made to the allocation(s). This procedure ensures that usage will only be permitted so long as the requestors have sufficient funds.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique lien identifier
<b>allocations</b>	<a href="#">Set&lt;LienAllocation&gt;</a>	The allocation amounts reserved with this lien.
<b>creationTime</b>	<b>Date</b>	The date this lien was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this lien is deleted or not
<b>description</b>	<b>String</b>	The lien description
<b>duration</b>	<b>Long</b>	The expected duration of the reserved usage in seconds
<b>endTime</b>	<b>Date</b>	The time the lien becomes inactive
<b>instance</b>	<b>String</b>	The lien is against the specified instance (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this lien was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the lien becomes active
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The id of the usage record associated with the lien and containing the usage properties

LienAllocation

Amounts of the allocations that the lien has holds against

Field Name	Type	Description
<b>id</b>	<b>String</b>	The child allocation id
<b>amount</b>	<b>Long</b>	The amount reserved against the allocation by this lien
<b>fund</b>	<b>Long</b>	The fund that the allocation is in
<b>lien</b>	<b>String</b>	The parent lien id

Related topics

- [Accounting Liens on page 1473](#)

**Fields: Organizations**

 See the associated [Accounting Organizations on page 1477](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/organizations	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.organizations.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Organization

An organization is a virtual organization in which accounts are grouped. An account may only belong to a single organization while an organization may have multiple accounts. For example, an account may represent a project or cost-center while an organization may represent an institutional department or business division. The purpose of defining organizations is to support the ability to produce reporting for higher-order organizational entities beyond the individual account. Default organization properties include an id (name in MAM) and a description. An organization can be created, queried, modified, and deleted.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique organization identifier
<b>creationTime</b>	<b>Date</b>	The date this organization was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this organization is deleted or not
<b>description</b>	<b>String</b>	The organization description
<b>modificationTime</b>	<b>Date</b>	The date this organization was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## API version 2

Organization

An organization is a virtual organization in which accounts are grouped. An account may only belong to a single organization while an organization may have multiple accounts. For example, an account may represent a project or cost-center while an organization may represent an institutional department or business division. The purpose of defining organizations is to support the ability to produce reporting for higher-order organizational entities beyond the individual account. Default organization properties include an id (name in MAM) and a description. An organization can be created, queried, modified, and deleted.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique organization identifier
<b>creationTime</b>	<b>Date</b>	The date this organization was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this organization is deleted or not
<b>description</b>	<b>String</b>	The organization description
<b>modificationTime</b>	<b>Date</b>	The date this organization was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

API version 1

Organization

An organization is a virtual organization in which accounts are grouped. An account may only belong to a single organization while an organization may have multiple accounts. For example, an account may represent a project or cost-center while an organization may represent an institutional department or business division. The purpose of defining organizations is to support the ability to produce reporting for higher-order organizational entities beyond the individual account. Default organization properties include an id (name in MAM) and a description. An organization can be created, queried, modified, and deleted.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique organization identifier
<b>creationTime</b>	<b>Date</b>	The date this organization was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this organization is deleted or not
<b>description</b>	<b>String</b>	The organization description
<b>modificationTime</b>	<b>Date</b>	The date this organization was last modified
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

Related topics

- [Accounting Organizations on page 1477](#)

**Fields: Quotes**

 See the associated [Accounting Quotes on page 1480](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/quotes	<a href="#">Permissions on page 1593</a>

Type	Value	Additional information
<b>Hooks filename</b>	accounting.quotes.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Quote

Quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique quote identifier
<b>amount</b>	<b>BigDecimal</b>	The total amount of the quote
<b>chargeRates</b>	<a href="#">Set&lt;QuoteChargeRate&gt;</a>	The applied charges that make up this quote.
<b>creationTime</b>	<b>Date</b>	The date this quote was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this quote is deleted or not
<b>description</b>	<b>String</b>	The quote description
<b>duration</b>	<b>Long</b>	The expected duration of the quoted usage in seconds
<b>endTime</b>	<b>Date</b>	The time the quote becomes inactive
<b>instance</b>	<b>String</b>	The quote instance name. (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this quote was last modified
<b>pinned</b>	<b>Boolean</b>	Boolean indicating whether the quote is pinned or not
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the quote becomes active

Field Name	Type	Description
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with this quote

### QuoteChargeRate

Saved charge rates to be used when the quote is referenced

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>name</b>	<b>String</b>	The child charge rate name
<b>quote</b>	<b>String</b>	The parent quote id
<b>value</b>	<b>String</b>	The child charge rate value

## API version 2

Quote

Quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique quote identifier
<b>amount</b>	<b>BigDecimal</b>	The total amount of the quote
<b>chargeRates</b>	<a href="#">Set&lt;QuoteChargeRate&gt;</a>	The applied charges that make up this quote.
<b>creationTime</b>	<b>Date</b>	The date this quote was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this quote is deleted or not
<b>description</b>	<b>String</b>	The quote description
<b>duration</b>	<b>Long</b>	The expected duration of the quoted usage in seconds
<b>endTime</b>	<b>Date</b>	The time the quote becomes inactive
<b>instance</b>	<b>String</b>	The quote instance name. (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this quote was last modified
<b>pinned</b>	<b>Boolean</b>	Boolean indicating whether the quote is pinned or not
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the quote becomes active

Field Name	Type	Description
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with this quote

### QuoteChargeRate

Saved charge rates to be used when the quote is referenced

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>name</b>	<b>String</b>	The child charge rate name
<b>quote</b>	<b>String</b>	The parent quote id
<b>value</b>	<b>String</b>	The child charge rate value

## API version 1

Quote

Quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique quote identifier
<b>amount</b>	<b>BigDecimal</b>	The total amount of the quote
<b>chargeRates</b>	<a href="#">Set&lt;QuoteChargeRate&gt;</a>	The applied charges that make up this quote.
<b>creationTime</b>	<b>Date</b>	The date this quote was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this quote is deleted or not
<b>description</b>	<b>String</b>	The quote description
<b>duration</b>	<b>Long</b>	The expected duration of the quoted usage in seconds
<b>endTime</b>	<b>Date</b>	The time the quote becomes inactive
<b>instance</b>	<b>String</b>	The quote instance name. (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	The date this quote was last modified
<b>pinned</b>	<b>Boolean</b>	Boolean indicating whether the quote is pinned or not
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>startTime</b>	<b>Date</b>	The time the quote becomes active

Field Name	Type	Description
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with this quote

QuoteChargeRate

Saved charge rates to be used when the quote is referenced

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>amount</b>	<b>String</b>	The charge rate amount
<b>name</b>	<b>String</b>	The child charge rate name
<b>quote</b>	<b>String</b>	The parent quote id
<b>value</b>	<b>String</b>	The child charge rate value

Related topics

- [Accounting Quotes on page 1480](#)

**Fields: Transactions**

 See the associated [Accounting Transactions on page 1483](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/transactions	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.transactions.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Transaction

Moab Accounting Manager logs all modifying transactions in a detailed transaction journal (queries are not recorded). Previous transactions can be queried but not modified or deleted. By default, a standard user may only query transactions performed by them.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique transaction identifier
<b>account</b>	<b>String</b>	The account name associated with the transaction
<b>action</b>	<b>String</b>	The transaction action name
<b>actor</b>	<b>String</b>	The authenticated user that performed the action
<b>allocation</b>	<b>Long</b>	The allocation id associated with the transaction
<b>amount</b>	<b>BigDecimal</b>	The amount
<b>child</b>	<b>String</b>	If the transaction object is an association, this is the value of the child
<b>creationTime</b>	<b>Date</b>	The date this transaction was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this transaction is deleted or not
<b>delta</b>	<b>BigDecimal</b>	The effective change (positive or negative) to the balance of an allocation
<b>description</b>	<b>String</b>	The description for the transaction
<b>duration</b>	<b>Long</b>	The duration associated with the transaction in seconds
<b>fund</b>	<b>Long</b>	The fund id associated with the transaction
<b>instance</b>	<b>String</b>	The instance name (e.g. the job id)
<b>key</b>	<b>String</b>	The object primary key value

Field Name	Type	Description
<b>machine</b>	<b>String</b>	The machine name associated with the transaction (e.g. the cluster name)
<b>modificationTime</b>	<b>Date</b>	The date this transaction was last modified
<b>object</b>	<b>String</b>	The transaction object name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with the transaction
<b>user</b>	<b>String</b>	The user name associated with the transaction

## API version 2

Transaction

Moab Accounting Manager logs all modifying transactions in a detailed transaction journal (queries are not recorded). Previous transactions can be queried but not modified or deleted. By default, a standard user may only query transactions performed by them.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique transaction identifier
<b>account</b>	<b>String</b>	The account name associated with the transaction
<b>action</b>	<b>String</b>	The transaction action name
<b>actor</b>	<b>String</b>	The authenticated user that performed the action
<b>allocation</b>	<b>Long</b>	The allocation id associated with the transaction
<b>amount</b>	<b>BigDecimal</b>	The amount
<b>child</b>	<b>String</b>	If the transaction object is an association, this is the value of the child
<b>creationTime</b>	<b>Date</b>	The date this transaction was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this transaction is deleted or not
<b>delta</b>	<b>BigDecimal</b>	The effective change (positive or negative) to the balance of an allocation
<b>description</b>	<b>String</b>	The description for the transaction
<b>duration</b>	<b>Long</b>	The duration associated with the transaction in seconds
<b>fund</b>	<b>Long</b>	The fund id associated with the transaction
<b>instance</b>	<b>String</b>	The instance name (e.g. the job id)
<b>key</b>	<b>String</b>	The object primary key value

Field Name	Type	Description
<b>machine</b>	<b>String</b>	The machine name associated with the transaction (e.g. the cluster name)
<b>modificationTime</b>	<b>Date</b>	The date this transaction was last modified
<b>object</b>	<b>String</b>	The transaction object name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with the transaction
<b>user</b>	<b>String</b>	The user name associated with the transaction

## API version 1

Transaction

Moab Accounting Manager logs all modifying transactions in a detailed transaction journal (queries are not recorded). Previous transactions can be queried but not modified or deleted. By default, a standard user may only query transactions performed by them.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	The unique transaction identifier
<b>account</b>	<b>String</b>	The account name associated with the transaction
<b>action</b>	<b>String</b>	The transaction action name
<b>actor</b>	<b>String</b>	The authenticated user that performed the action
<b>allocation</b>	<b>Long</b>	The allocation id associated with the transaction
<b>amount</b>	<b>BigDecimal</b>	The amount
<b>child</b>	<b>String</b>	If the transaction object is an association, this is the value of the child
<b>creationTime</b>	<b>Date</b>	The date this transaction was created
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this transaction is deleted or not
<b>delta</b>	<b>BigDecimal</b>	The effective change (positive or negative) to the balance of an allocation
<b>description</b>	<b>String</b>	The description for the transaction
<b>duration</b>	<b>Long</b>	The duration associated with the transaction in seconds
<b>fund</b>	<b>Long</b>	The fund id associated with the transaction
<b>instance</b>	<b>String</b>	The instance name (e.g. the job id)
<b>key</b>	<b>String</b>	The object primary key value

Field Name	Type	Description
<b>machine</b>	<b>String</b>	The machine name associated with the transaction (e.g. the cluster name)
<b>modificationTime</b>	<b>Date</b>	The date this transaction was last modified
<b>object</b>	<b>String</b>	The transaction object name
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction
<b>usageRecord</b>	<b>Long</b>	The usage record id associated with the transaction
<b>user</b>	<b>String</b>	The user name associated with the transaction

Related topics

- [Accounting Transactions on page 1483](#)

**Fields: Usage Records**

**i** See the associated [Accounting Usage records on page 1488](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/usage-records	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.usage-record-s.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

UsageRecord

A usage record tracks the usage of resources and services on your system, recording the charge and the details of the usage in a usage record.

Usage Record quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

A usage lien can be used to place a hold on the user's fund before usage starts to ensure that the credits will be there when it completes. The replace option may be specified if you want the new lien to replace existing liens of the same instance name (associated with the same usage record). The modify option may be specified to dynamically extend any existing lien with the same instance name with the specified characteristics instead of creating a new one.

A usage charge debits the appropriate allocations based on the attributes of the usage. The charge is calculated based on factors including the resources and services used, the usage time, and other quality-based factors. By default, any liens associated with the charge will be removed. The incremental option may be specified if you want associated liens to be reduced instead of removed. If a usage record already exists for the instance being charged it will be updated with the data properties passed in with the charge request, otherwise a new usage record will be created.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	The unique usage record identifier
<b>charge</b>	<b>String</b>	No	The cumulative amount charged
<b>creationTime</b>	<b>Date</b>	No	The date this usage record was created
<b>deleted</b>	<b>Boolean</b>	No	A boolean indicating whether this usage record is deleted or not
<b>instance</b>	<b>String</b>	No	The usage record instance name (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	No	The date this usage record was last modified

Field Name	Type	POST	Description
<b>qualityOfService</b>	<b>String</b>	No	The quality of service associated with the usage
<b>quote</b>	<b>Long</b>	No	The associated quote id
<b>requestId</b>	<b>Long</b>	No	The id of the last modifying request
<b>stage</b>	<b>String</b>	No	The last affecting action (i.e. Create, Quote, Reserve, Query)
<b>transactionId</b>	<b>Long</b>	No	The id of the last modifying transaction
<b>type</b>	<b>String</b>	No	The usage record type
<b>user</b>	<b>String</b>	No	The user name associated with the usage

## API version 2

UsageRecord

A usage record tracks the usage of resources and services on your system, recording the charge and the details of the usage in a usage record.

Usage Record quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

A usage lien can be used to place a hold on the user's fund before usage starts to ensure that the credits will be there when it completes. The replace option may be specified if you want the new lien to replace existing liens of the same instance name (associated with the same usage record). The modify option may be specified to dynamically extend any existing lien with the same instance name with the specified characteristics instead of creating a new one.

A usage charge debits the appropriate allocations based on the attributes of the usage. The charge is calculated based on factors including the resources and services used, the usage time, and other quality-based factors. By default, any liens associated with the charge will be removed. The incremental option may be specified if you want associated liens to be reduced instead of removed. If a usage record already exists for the instance being charged it will be updated with the data properties passed in with the charge request, otherwise a new usage record will be created.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	The unique usage record identifier
<b>charge</b>	<b>String</b>	No	The cumulative amount charged
<b>creationTime</b>	<b>Date</b>	No	The date this usage record was created
<b>deleted</b>	<b>Boolean</b>	No	A boolean indicating whether this usage record is deleted or not
<b>instance</b>	<b>String</b>	No	The usage record instance name (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	No	The date this usage record was last modified

Field Name	Type	POST	Description
<b>qualityOfService</b>	<b>String</b>	No	The quality of service associated with the usage
<b>quote</b>	<b>Long</b>	No	The associated quote id
<b>requestId</b>	<b>Long</b>	No	The id of the last modifying request
<b>stage</b>	<b>String</b>	No	The last affecting action (i.e. Create, Quote, Reserve, Query)
<b>transactionId</b>	<b>Long</b>	No	The id of the last modifying transaction
<b>type</b>	<b>String</b>	No	The usage record type
<b>user</b>	<b>String</b>	No	The user name associated with the usage

## API version 1

UsageRecord

A usage record tracks the usage of resources and services on your system, recording the charge and the details of the usage in a usage record.

Usage Record quotes can be used to determine how much it will cost to use a resource or service. Provided the cost-only option is not specified, this step will additionally verify that the submitter has sufficient funds and meets all the allocation policy requirements for the usage, and can be used at the submission of the usage request as an early filter to prevent the usage from getting blocked when it tries to obtain a lien to start later. If a guaranteed quote is requested, a quote id is returned and can be used in the subsequent charge to guarantee the rates that were used to form the original quote. A guaranteed quote has the side effect of creating a quote record and a permanent usage record. A quote id will be returned which can be used with the lien and charge to claim the quoted charge rates. A cost-only quote can be used to determine how much would be charged for usage without verifying sufficient funds or checking to see if the charge could succeed.

A usage lien can be used to place a hold on the user's fund before usage starts to ensure that the credits will be there when it completes. The replace option may be specified if you want the new lien to replace existing liens of the same instance name (associated with the same usage record). The modify option may be specified to dynamically extend any existing lien with the same instance name with the specified characteristics instead of creating a new one.

A usage charge debits the appropriate allocations based on the attributes of the usage. The charge is calculated based on factors including the resources and services used, the usage time, and other quality-based factors. By default, any liens associated with the charge will be removed. The incremental option may be specified if you want associated liens to be reduced instead of removed. If a usage record already exists for the instance being charged it will be updated with the data properties passed in with the charge request, otherwise a new usage record will be created.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	The unique usage record identifier
<b>charge</b>	<b>String</b>	No	The cumulative amount charged
<b>creationTime</b>	<b>Date</b>	No	The date this usage record was created
<b>deleted</b>	<b>Boolean</b>	No	A boolean indicating whether this usage record is deleted or not
<b>instance</b>	<b>String</b>	No	The usage record instance name (i.e. job id)
<b>modificationTime</b>	<b>Date</b>	No	The date this usage record was last modified

Field Name	Type	POST	Description
<b>qualityOfService</b>	<b>String</b>	No	The quality of service associated with the usage
<b>quote</b>	<b>Long</b>	No	The associated quote id
<b>requestId</b>	<b>Long</b>	No	The id of the last modifying request
<b>stage</b>	<b>String</b>	No	The last affecting action (i.e. Create, Quote, Reserve, Query)
<b>transactionId</b>	<b>Long</b>	No	The id of the last modifying transaction
<b>type</b>	<b>String</b>	No	The usage record type
<b>user</b>	<b>String</b>	No	The user name associated with the usage

#### Related topics

- [Accounting Usage records on page 1488](#)

#### Fields: Users

**i** See the associated [Accounting Users on page 1502](#) resource section for more information on how to use this resource and supported operations.

#### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	accounting/users	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	accounting.users.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

User

A user is a person authorized to use a resource or service. Default user properties include the common name, phone number, email address, default account, and description for that person.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>creationTime</b>	<b>Date</b>	The date this user was created
<b>defaultAccount</b>	<b>String</b>	The default account for this user
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this user is deleted or not
<b>description</b>	<b>String</b>	The user description
<b>emailAddress</b>	<b>String</b>	The user's email address
<b>modificationTime</b>	<b>Date</b>	The date this user was last modified
<b>phoneNumber</b>	<b>String</b>	The user's phone number
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## API version 2

User

A user is a person authorized to use a resource or service. Default user properties include the common name, phone number, email address, default account, and description for that person.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>creationTime</b>	<b>Date</b>	The date this user was created
<b>defaultAccount</b>	<b>String</b>	The default account for this user
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this user is deleted or not
<b>description</b>	<b>String</b>	The user description
<b>emailAddress</b>	<b>String</b>	The user's email address
<b>modificationTime</b>	<b>Date</b>	The date this user was last modified
<b>phoneNumber</b>	<b>String</b>	The user's phone number
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## API version 1

User

A user is a person authorized to use a resource or service. Default user properties include the common name, phone number, email address, default account, and description for that person.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique user identifier
<b>active</b>	<b>Boolean</b>	A boolean indicating whether this user is active or not
<b>creationTime</b>	<b>Date</b>	The date this user was created
<b>defaultAccount</b>	<b>String</b>	The default account for this user
<b>deleted</b>	<b>Boolean</b>	A boolean indicating whether this user is deleted or not
<b>description</b>	<b>String</b>	The user description
<b>emailAddress</b>	<b>String</b>	The user's email address
<b>modificationTime</b>	<b>Date</b>	The date this user was last modified
<b>phoneNumber</b>	<b>String</b>	The user's phone number
<b>requestId</b>	<b>Long</b>	The id of the last modifying request
<b>transactionId</b>	<b>Long</b>	The id of the last modifying transaction

## Related topics

- [Accounting Users on page 1502](#)

## Fields: Credentials

 See the associated [Credentials on page 1506](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	credentials	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	credentials.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

API version 3

Credential

A credential is an entity, such as a user or a group, that has access to resources. Credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	PUT	Description
<b>name</b>	<b>String</b>	No	The name of the credential.

API version 2

Credential

A credential is an entity, such as a user or a group, that has access to resources. Credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	PUT	Description
<b>name</b>	<b>String</b>	No	The name of the credential.

Related topics

- [Credentials on page 1506](#)

Fields: Events

 See the associated [Events on page 1528](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	events	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	events.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

API version 3

Event

Represents an event originating from any component in the system (MWM, MWS, MAM, etc). Events are related to, but not the same as, [Notifications](#). See [NotificationCondition](#) for an explanation of when to use an event vs a notification.

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique ID for this event
<b>arguments</b>	<b>List&lt;String&gt;</b>	Yes	The event's arguments
<b>associatedObjects</b>	<a href="#">Set&lt;AssociatedObject&gt;</a>	Yes	Objects relating to the event
<b>code</b>	<b>int</b>	Yes	This is a positive, 32-bit numeric value. Source code that needs to take action on events based on which event (error) occurred can switch based on this value. The top 16 bits are determined by the severity of the event and the component that emits it. The bottom 16 bits are assigned by any arbitrary mechanism convenient to a component. Each component thus has 64k unique event codes that it can assign. Once assigned, event codes are immutable; it can never be the case that error 12345 means one thing in release A, and a different thing in release B.
<b>eventDate</b>	<b>Date</b>	Yes	The date and time the event occurred, not the date and time MWS received the event. It is up to the reporting component to report this time accurately. Required during POST.
<b>eventType</b>	<b>String</b>	Yes	Signifies what type of event. Cannot contain single quotes(') or double quotes(").
<b>message</b>	<b>String</b>	Yes	A summary of what happened that caused this event
<b>origin</b>	<b>String</b>	Yes	The origin of this event. Cannot contain single quotes(') or double quotes(").
<b>severity</b>	<a href="#">EventSeverity</a>	Yes	Signifies the severity of an event.

Field Name	Type	POST	Description
<b>tenant</b>	<b>Map&lt;String, String&gt;</b>	No	The event's tenant (contains tenant id and name)

AssociatedObject

Represents and uniquely identifies an object associated with an event. (e.g node, job, reservation, trigger)

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	Yes	The object id (e.g. reservation.1, job.21, vm3). Cannot contain single quotes(') or double quotes(").
<b>type</b>	<b>String</b>	Yes	The type of object (e.g. node, job, reservation). Cannot contain single quotes(') or double quotes(").

EventSeverity

Value	Description
<b>INFO</b>	
<b>WARN</b>	
<b>ERROR</b>	
<b>FATAL</b>	

API version 2

EventVersion2

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique ID for this event
<b>details</b>	<b>Map&lt;String, Map&gt;</b>	Yes	A map where detail name maps to detail value. (e.g. "sourceHypervisor" => "blade256", "destinationHypervisor" => "blade257", "os" => "centos-5.5-stateless")
<b>errorMessage</b>	<a href="#"><u>ErrorMessageVersion2</u></a>	Yes	Details about any errors associated with the event. If this event was not associated with any errors this field will be null
<b>eventCategory</b>	<b>String</b>	Yes	Signifies what category of event.
<b>eventTime</b>	<b>Date</b>	Yes	The time the event occurred, not the time MWS received the event. It is up to the reporting component to report this time accurately. Corresponds to eventDate in API Version 3. Required during POST.
<b>eventType</b>	<b>String</b>	Yes	Signifies what type of event.
<b>facility</b>	<b>String</b>	Yes	A categorization of how this event fits in with other events.
<b>initiatedBy</b>	<a href="#"><u>UserDetailsVersion2</u></a>	Yes	Details about the user that initiated this event
<b>primaryObject</b>	<a href="#"><u>MoabObjectVersion2</u></a>	Yes	Most events will have a "primary object" associated with it. An event can have at most ONE primary object. For example, a JobStart event will have a primary job object, so the type would be "job" and the object ID would be the ID of the job. Primary objects are, however, optional, depending on the type of event. For example, a "SchedulerCommand" event does not have a primary object.

Field Name	Type	POST	Description
<b>relatedObjects</b>	<a href="#">Set&lt;MoabObjectVersion2&gt;</a>	Yes	Objects relating to the event that are not the primary object. Corresponds to associatedObjects in API Version 3.
<b>severity</b>	<b>String</b>	Yes	Signifies the severity of an event. Severity can be "FATAL", "ERROR", "WARN", "INFO"
<b>sourceComponent</b>	<b>String</b>	Yes	What Adaptive Computing component reported this event. Examples: "MWM", "MWS", "MAM", etc. Corresponds to origin in API Version 3.
<b>status</b>	<b>String</b>	Yes	The status of the reported event.

[ErrorMessageVersion2](#)

Field Name	Type	POST	Description
<b>errorCode</b>	<b>String</b>	Yes	The original error code generated or detected by the originator.
<b>message</b>	<b>String</b>	Yes	If an event has a status of "failure" or other non-successful operation, this field should provide a human-friendly error message Corresponds to Event.message in API Version 3 and above.
<b>originator</b>	<b>String</b>	Yes	The software component or entity that generated or detected the error (e.g. Moab, Torque, MWS, Viewpoint, RM, Database, etc).

[UserDetailsVersion2](#)

Field Name	Type	POST	Description
<b>proxyUser</b>	<b>String</b>	Yes	The proxy user that initiated the event.
<b>user</b>	<b>String</b>	Yes	The user that initiated the event.

[MoabObjectVersion2](#)

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	Yes	The moab object id (e.g. reservation.1, job.21, vm3)
<b>serialization</b>	<b>String</b>	Yes	A serialized representation of the object
<b>type</b>	<b>String</b>	Yes	The moab object type (e.g. node, job, reservation)

## API version 1

EventVersion2

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique ID for this event
<b>details</b>	<b>Map&lt;String, Map&gt;</b>	Yes	A map where detail name maps to detail value. (e.g. "sourceHypervisor" => "blade256", "destinationHypervisor" => "blade257", "os" => "centos-5.5-stateless")
<b>errorMessage</b>	<a href="#"><u>ErrorMessageVersion2</u></a>	Yes	Details about any errors associated with the event. If this event was not associated with any errors this field will be null
<b>eventCategory</b>	<b>String</b>	Yes	Signifies what category of event.
<b>eventTime</b>	<b>Date</b>	Yes	The time the event occurred, not the time MWS received the event. It is up to the reporting component to report this time accurately. Corresponds to eventDate in API Version 3. Required during POST.
<b>eventType</b>	<b>String</b>	Yes	Signifies what type of event.
<b>facility</b>	<b>String</b>	Yes	A categorization of how this event fits in with other events.
<b>initiatedBy</b>	<a href="#"><u>UserDetailsVersion2</u></a>	Yes	Details about the user that initiated this event
<b>primaryObject</b>	<a href="#"><u>MoabObjectVersion2</u></a>	Yes	Most events will have a "primary object" associated with it. An event can have at most ONE primary object. For example, a JobStart event will have a primary job object, so the type would be "job" and the object ID would be the ID of the job. Primary objects are, however, optional, depending on the type of event. For example, a "SchedulerCommand" event does not have a primary object.

Field Name	Type	POST	Description
<b>relatedObjects</b>	<a href="#">Set&lt;MoabObjectVersion2&gt;</a>	Yes	Objects relating to the event that are not the primary object. Corresponds to associatedObjects in API Version 3.
<b>severity</b>	<b>String</b>	Yes	Signifies the severity of an event. Severity can be "FATAL", "ERROR", "WARN", "INFO"
<b>sourceComponent</b>	<b>String</b>	Yes	What Adaptive Computing component reported this event. Examples: "MWM", "MWS", "MAM", etc. Corresponds to origin in API Version 3.
<b>status</b>	<b>String</b>	Yes	The status of the reported event.

#### [ErrorMessageVersion2](#)

Field Name	Type	POST	Description
<b>errorCode</b>	<b>String</b>	Yes	The original error code generated or detected by the originator.
<b>message</b>	<b>String</b>	Yes	If an event has a status of "failure" or other non-successful operation, this field should provide a human-friendly error message Corresponds to Event.message in API Version 3 and above.
<b>originator</b>	<b>String</b>	Yes	The software component or entity that generated or detected the error (e.g. Moab, Torque, MWS, Viewpoint, RM, Database, etc).

#### [UserDetailsVersion2](#)

Field Name	Type	POST	Description
<b>proxyUser</b>	<b>String</b>	Yes	The proxy user that initiated the event.
<b>user</b>	<b>String</b>	Yes	The user that initiated the event.

#### [MoabObjectVersion2](#)

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	Yes	The moab object id (e.g. reservation.1, job.21, vm3)
<b>serialization</b>	<b>String</b>	Yes	A serialized representation of the object
<b>type</b>	<b>String</b>	Yes	The moab object type (e.g. node, job, reservation)

#### Related topics

- [Events on page 1528](#)

## Fields: Images

**i** See the associated [Images on page 1536](#) resource section for more information on how to use this resource and supported operations.

#### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	images	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	images.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

Image

An image is used to track the different types of operating systems and hypervisors available in a data center. If the image is a hypervisor, it can contain other images which are the available virtual machines of the hypervisor.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this image.
<b>active</b>	<b>Boolean</b>	Yes	Yes	If false, the image is flagged as inactive and should not be used. Defaults to true.

Field Name	Type	POST	PUT	Description
<b>extensions</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	<p>A map containing maps which represent settings for provisioning managers. Only one extension may be present on an image at a time currently. Valid default provisioning manager specific extensions include 'xcat'.</p> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is false:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> </ul> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is true:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> <li>• hvGroupName - The name of the xCAT hypervisor group</li> <li>• vmGroupName - The name of the xCAT VM group</li> </ul>
<b>features</b>	<b>Set&lt;String&gt;</b>	Yes	Yes	The set of features used by the provisioning manager.

Field Name	Type	POST	PUT	Description
<b>hypervisor</b>	<b>Boolean</b>	Yes	Yes	Whether or not the image is a hypervisor. Required during POST. Note that this is related to, but not the same as, <a href="#">supportsPhysicalMachine</a> . Also, when this is false, no <a href="#">virtualizedImages</a> may be specified for an image.
<b>hypervisorType</b>	<b>String</b>	Yes	Yes	The type of the hypervisor, which is indicative of the hypervisor technology used in this image. Required if this image is a hypervisor image.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this image. Required during POST.
<b>osType</b>	<b>String</b>	Yes	Yes	The type of the operating system such as 'Linux' or 'Windows'. Required during POST.
<b>supportsPhysicalMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a physical machine, defaults to false. Either this or <a href="#">supportsVirtualMachine</a> must be set to true. Note that this is related to, but not the same as, <a href="#">hypervisor</a> . Some images may not be hypervisors but can be provisioned on a physical machine.
<b>supportsVirtualMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a virtual machine, defaults to false. Either this or <a href="#">supportsPhysicalMachine</a> must be set to true.
<b>templateName</b>	<b>String</b>	Yes	Yes	The VM template to use for this image. Only valid if the <a href="#">type</a> is set to a valid template type such as 'ImageType.LINKED_CLONE'.

Field Name	Type	POST	PUT	Description
<b>type</b>	<a href="#">ImageType</a>	Yes	Yes	The type of the image. This property may affect the valid values to use for other fields. See <a href="#">ImageType</a> for more information. (See also: <a href="#">templateName</a> .)
<b>virtualizedImages</b>	<a href="#">Set&lt;Image&gt;</a>	Yes	Yes	The set of images available on this hypervisor.

[ImageType](#)

Represents an image type, such as stateful or stateless. This is used by provisioning managers and applications to correctly provision and represent the image.

Certain types are only valid for images configured as templates using the [Image.templateName](#) field. This currently includes [ImageType.LINKED\\_CLONE](#) and [ImageType.FULL\\_CLONE](#).

Value	Description
<b>STATEFUL</b>	
<b>STATELESS</b>	
<b>STATELITE</b>	
<b>LINKED_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.
<b>FULL_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.

## API version 2

Image

An image is used to track the different types of operating systems and hypervisors available in a data center. If the image is a hypervisor, it can contain other images which are the available virtual machines of the hypervisor.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this image.
<b>active</b>	<b>Boolean</b>	Yes	Yes	If false, the image is flagged as inactive and should not be used. Defaults to true.

Field Name	Type	POST	PUT	Description
<b>extensions</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	<p>A map containing maps which represent settings for provisioning managers. Only one extension may be present on an image at a time currently. Valid default provisioning manager specific extensions include 'xcat'.</p> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is false:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> </ul> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is true:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> <li>• hvGroupName - The name of the xCAT hypervisor group</li> <li>• vmGroupName - The name of the xCAT VM group</li> </ul>
<b>features</b>	<b>Set&lt;String&gt;</b>	Yes	Yes	The set of features used by the provisioning manager.

Field Name	Type	POST	PUT	Description
<b>hypervisor</b>	<b>Boolean</b>	Yes	Yes	Whether or not the image is a hypervisor. Required during POST. Note that this is related to, but not the same as, <a href="#">supportsPhysicalMachine</a> . Also, when this is false, no <a href="#">virtualizedImages</a> may be specified for an image.
<b>hypervisorType</b>	<b>String</b>	Yes	Yes	The type of the hypervisor, which is indicative of the hypervisor technology used in this image. Required if this image is a hypervisor image.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this image. Required during POST.
<b>osType</b>	<b>String</b>	Yes	Yes	The type of the operating system such as 'Linux' or 'Windows'. Required during POST.
<b>supportsPhysicalMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a physical machine, defaults to false. Either this or <a href="#">supportsVirtualMachine</a> must be set to true. Note that this is related to, but not the same as, <a href="#">hypervisor</a> . Some images may not be hypervisors but can be provisioned on a physical machine.
<b>supportsVirtualMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a virtual machine, defaults to false. Either this or <a href="#">supportsPhysicalMachine</a> must be set to true.
<b>templateName</b>	<b>String</b>	Yes	Yes	The VM template to use for this image. Only valid if the <a href="#">type</a> is set to a valid template type such as 'ImageType.LINKED_CLONE'.

Field Name	Type	POST	PUT	Description
<b>type</b>	<a href="#">ImageType</a>	Yes	Yes	The type of the image. This property may affect the valid values to use for other fields. See <a href="#">ImageType</a> for more information. (See also: <a href="#">templateName</a> .)
<b>virtualizedImages</b>	<a href="#">Set&lt;Image&gt;</a>	Yes	Yes	The set of images available on this hypervisor.

[ImageType](#)

Represents an image type, such as stateful or stateless. This is used by provisioning managers and applications to correctly provision and represent the image.

Certain types are only valid for images configured as templates using the [Image.templateName](#) field. This currently includes [ImageType.LINKED\\_CLONE](#) and [ImageType.FULL\\_CLONE](#).

Value	Description
<b>STATEFUL</b>	
<b>STATELESS</b>	
<b>STATELITE</b>	
<b>LINKED_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.
<b>FULL_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.

## API version 1

Image

An image is used to track the different types of operating systems and hypervisors available in a data center. If the image is a hypervisor, it can contain other images which are the available virtual machines of the hypervisor.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this image.
<b>active</b>	<b>Boolean</b>	Yes	Yes	If false, the image is flagged as inactive and should not be used. Defaults to true.

Field Name	Type	POST	PUT	Description
<b>extensions</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	<p>A map containing maps which represent settings for provisioning managers. Only one extension may be present on an image at a time currently. Valid default provisioning manager specific extensions include 'xcat'.</p> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is false:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> </ul> <p>Required properties for 'xcat' when <a href="#">hypervisor</a> is true:</p> <ul style="list-style-type: none"> <li>• os - The name of the operating system according to xCAT</li> <li>• architecture - The architecture, such as x86_64</li> <li>• profile - The xCAT profile to use for the image</li> <li>• hvGroupName - The name of the xCAT hypervisor group</li> <li>• vmGroupName - The name of the xCAT VM group</li> </ul>
<b>features</b>	<b>Set&lt;String&gt;</b>	Yes	Yes	The set of features used by the provisioning manager.

Field Name	Type	POST	PUT	Description
<b>hypervisor</b>	<b>Boolean</b>	Yes	Yes	Whether or not the image is a hypervisor. Required during POST. Note that this is related to, but not the same as, <a href="#">supportsPhysicalMachine</a> . Also, when this is false, no <a href="#">virtualizedImages</a> may be specified for an image.
<b>hypervisorType</b>	<b>String</b>	Yes	Yes	The type of the hypervisor, which is indicative of the hypervisor technology used in this image. Required if this image is a hypervisor image.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this image. Required during POST.
<b>osType</b>	<b>String</b>	Yes	Yes	The type of the operating system such as 'Linux' or 'Windows'. Required during POST.
<b>supportsPhysicalMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a physical machine, defaults to false. Either this or <a href="#">supportsVirtualMachine</a> must be set to true. Note that this is related to, but not the same as, <a href="#">hypervisor</a> . Some images may not be hypervisors but can be provisioned on a physical machine.
<b>supportsVirtualMachine</b>	<b>Boolean</b>	Yes	Yes	Specifies whether the image can be used to provision a virtual machine, defaults to false. Either this or <a href="#">supportsPhysicalMachine</a> must be set to true.
<b>templateName</b>	<b>String</b>	Yes	Yes	The VM template to use for this image. Only valid if the <a href="#">type</a> is set to a valid template type such as 'ImageType.LINKED_CLONE'.

Field Name	Type	POST	PUT	Description
<b>type</b>	<a href="#">ImageType</a>	Yes	Yes	The type of the image. This property may affect the valid values to use for other fields. See <a href="#">ImageType</a> for more information. (See also: <a href="#">templateName</a> .)
<b>virtualizedImages</b>	<a href="#">Set&lt;Image&gt;</a>	Yes	Yes	The set of images available on this hypervisor.

### [ImageType](#)

Represents an image type, such as stateful or stateless. This is used by provisioning managers and applications to correctly provision and represent the image.

Certain types are only valid for images configured as templates using the [Image.templateName](#) field. This currently includes [ImageType.LINKED\\_CLONE](#) and [ImageType.FULL\\_CLONE](#).

Value	Description
<b>STATEFUL</b>	
<b>STATELESS</b>	
<b>STATELITE</b>	
<b>LINKED_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.
<b>FULL_CLONE</b>	Template type. When this image type is used, the <a href="#">Image.hypervisor</a> field must be set to false, <a href="#">Image.supportsVirtualMachine</a> must be true, and <a href="#">Image.supportsPhysicalMachine</a> must be false.

### Related topics

- [Images](#) on page 1536

### Fields: Job Arrays

**i** See the associated [Job arrays](#) on page 1545 resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	job-arrays	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	job-arrays.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

JobArray

Job arrays are an easy way to submit many sub-jobs that perform the same work using the same script, but operate on different sets of data. Sub-jobs are the jobs created by an array job and are identified by the array job ID and an index; for example, if 235[1] is an identifier, the number 235 is a job array ID, and 1 is the sub-job.

Field Name	Type	POST	Description
<b>cancellationPolicy</b>	<a href="#">CancellationPolicyInformation</a>	Yes	Represents the cancellation policy to use for the job array.
<b>indexRanges</b>	<a href="#">List&lt;JobArrayIndexRange&gt;</a>	Yes	The index ranges used to generate the sub-job indices. To use hard-coded values, see <a href="#">indexValues</a> .
<b>indexValues</b>	<b>List&lt;Long&gt;</b>	Yes	The index values to use for the sub-jobs. To use ranges, see <a href="#">indexRanges</a> .
<b>jobPrototype</b>	<a href="#">Job</a>	Yes	The definition of the job to use for each sub-job.
<b>name</b>	<b>String</b>	Yes	The name of the job array. In MWS API version 1, this is stored in the <code>name</code> field of the created jobs. In MWS API version 2, this is stored in the <code>customName</code> field of the created jobs.
<b>slotLimit</b>	<b>Long</b>	Yes	(Optional) The number of sub-jobs in the array that can run at a time.

CancellationPolicyInformation

Job arrays can be canceled based on the success or failure of the first or any sub-job. This class represents the failure policies.

Field Name	Type	POST	Description
<b>anyJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of any sub-job. May be used in combination with <a href="#">firstJob</a> .
<b>firstJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of the first sub-job (array index 1). May be used in combination with <a href="#">anyJob</a> .

CancellationPolicy

This enumeration represents job array cancellation policies, and is to be used in combination with CancellationPolicyInformation.

Value	Description
<b>SUCCESS</b>	Cancels the job array if the specified sub-job succeeds.
<b>FAILURE</b>	Cancels the job array if the specified sub-job fails.

JobArrayIndexRange

Represents information about a job index expression. This is used when creating job arrays only.

Field Name	Type	POST	Description
<b>endIndex</b>	<b>Long</b>	Yes	The end of the index range. i.e. 10 for 1-10.
<b>increment</b>	<b>Long</b>	Yes	The increment of the index range, defaults to 1 and must be greater than 0. For a range of 1-10 with an increment of 2, the list of indices will be [1, 3, 5, 7, 9].
<b>startIndex</b>	<b>Long</b>	Yes	The start of the index range. i.e. 1 for 1-10.

Job

This class represents a job in the Moab Workload Manager. A job is a request for compute resources (CPUs, memory, storage) with which the requester can do work for a given amount of time. In an HPC environment, this might be a batch script to perform a Monte Carlo simulation. In a cloud environment, this would be a virtual machine and its associated storage. Moab will evaluate the request and assign the requested resources to the requester based on policies, current demand, and other factors in the data center. A job will also usually have some process that Moab starts automatically at the assigned start time. In an HPC environment, this can be starting a batch script on the assigned nodes. In a cloud environment, this can be starting provisioning processes to create the virtual machine and storage and install software on it.

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier of this job. Note: this field is not user-assigned and is generated by the database.

Field Name	Type	POST	Description
<b>arrayIndex</b>	<b>Long</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>attributes</b>	<b>Set&lt;String&gt;</b>	Yes	The list of generic attributes associated with this job.
<b>blocks</b>	<a href="#">Set&lt;JobBlock&gt;</a>	No	Reasons the job is blocked from running.
<b>bypassCount</b>	<b>Integer</b>	No	The number of times the job has been backfilled.
<b>cancelCount</b>	<b>Integer</b>	No	The number of times a job has received a cancel request.
<b>commandFile</b>	<b>String</b>	Yes	The path to the file that is executed when this job runs. This is the script that will call all the work of this job.
<b>commandLineArguments</b>	<b>String</b>	Yes	The command line arguments passed to the <code>commandFile</code> .
<b>completionCode</b>	<b>Integer</b>	No	The exit code from this job.
<b>cpuTime</b>	<b>Long</b>	No	CPU usage time in seconds as reported by the resource manager.

Field Name	Type	POST	Description
<b>credentials</b>	<a href="#">JobCredentials</a>	Yes	The credentials (user and group, for example) associated with this job.
<b>customName</b>	<b>String</b>	Yes	The user-specified name of this job.
<b>dates</b>	<a href="#">JobDates</a>	Yes	Various dates associated with this job.
<b>deferCount</b>	<b>Integer</b>	No	The number of times a job has been deferred.
<b>dependencies</b>	<a href="#">Set&lt;JobDependency&gt;</a>	Yes	Dependencies that must be fulfilled before the job can start.
<b>description</b>	<b>String</b>	No	The description of the job. Can be set only in a job template.
<b>duration</b>	<b>Long</b>	Yes	The length of time in seconds requested for the job. Note that it is possible to set duration to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>durationActive</b>	<b>Long</b>	No	The length of time in seconds the job has been active or running.
<b>durationQueued</b>	<b>Long</b>	No	The length of time in seconds the job has been eligible to run in the queue.
<b>durationRemaining</b>	<b>Long</b>	No	An estimate of the time remaining, in seconds, before the job will complete.
<b>durationSuspended</b>	<b>Long</b>	No	The length of time in seconds the job has been suspended.

Field Name	Type	POST	Description
<b>emailNotifyAddresses</b>	<b>Set&lt;String&gt;</b>	Yes	The list of addresses to whom email is sent by the execution server.
<b>emailNotifyTypes</b>	<a href="#"><u>Set&lt;JobEmailNotifyType&gt;</u></a>	Yes	The list of email notify types attached to the job.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	Setting this field to true tells the Moab Workload Manager to set various variables, if populated, in the job's environment.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of environment variables for this job.
<b>epilogScript</b>	<b>String</b>	Yes	The path to the TORQUE epilog script.
<b>flags</b>	<a href="#"><u>Set&lt;JobFlag&gt;</u></a>	Yes	The flags that are set on this job.
<b>holdDate</b>	<b>Date</b>	No	The date the most recent hold was placed on the job.
<b>holdReason</b>	<a href="#"><u>JobHoldReason</u></a>	No	The reason the job is on hold.
<b>holds</b>	<a href="#"><u>Set&lt;JobHoldType&gt;</u></a>	Yes	The holds that are set on the job. The "User" hold type is valid during POST.
<b>initialWorkingDirectory</b>	<b>String</b>	Yes	The path to the directory in which the job will be started.
<b>isActive</b>	<b>Boolean</b>	No	True if the job is active, false if the job is complete.
<b>jobGroup</b>	<b>String</b>	Yes	The job group to which this job belongs (different from credentials.group).

Field Name	Type	POST	Description
<b>masterNode</b>	<a href="#">DomainProxy</a>	No	The first node in the list of allocated nodes for this job. For TORQUE jobs, this represents the "mother superior."
<b>memorySecondsDedicated</b>	Double	No	The memory seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>memorySecondsUtilized</b>	Double	No	The memory seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	No	The list of messages associated with the job. The "message" field is valid during PUT.
<b>migrateCount</b>	Integer	No	The number of times the job has been migrated.
<b>minimumPreemptTime</b>	Long	No	The minimum length of time, in seconds, an active job must be running before it is eligible for preemption.
<b>mwmName</b>	String	No	The name of the Moab Workload Manager instance that owns this job.
<b>name</b>	String	No	The name of this job. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>nodesExcluded</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The list of nodes that should not be considered for this job.

Field Name	Type	POST	Description
<b>nodesRequested</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The exact set, superset, or subset of nodes on which this job must run. (See also: <a href="#">nodesRequestedPolicy</a> .)
<b>nodesRequestedPolicy</b>	<a href="#">JobHostListMode</a>	Yes	Indicates an exact set, superset, or subset of nodes on which the job must run. Only relevant if nodesRequested is provided. (See also: <a href="#">nodesRequested</a> .)
<b>partitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	The list of partitions that this job can access.
<b>partitionAccessListRequested</b>	<b>Set&lt;String&gt;</b>	Yes	The list of partitions that this job has requested.
<b>preemptCount</b>	<b>Integer</b>	No	The number of times the job has been preempted.
<b>priorities</b>	<a href="#">JobPriority</a>	Yes	The list of priorities for the job.
<b>processorSecondsDedicated</b>	<b>Double</b>	No	The processor seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>processorSecondsLimit</b>	<b>Double</b>	No	The limit for processorSecondsUtilized.
<b>processorSecondsUtilized</b>	<b>Double</b>	No	The processor seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>prologScript</b>	<b>String</b>	Yes	The path to the TORQUE prolog script.

Field Name	Type	POST	Description
<b>queueStatus</b>	<a href="#">JobQueueStatus</a>	No	The status of the job in its queue.
<b>rejectPolicies</b>	<a href="#">Set&lt;JobRejectPolicy&gt;</a>	No	The list of policies enabled when a job is rejected.
<b>requirements</b>	<a href="#">Set&lt;JobRequirement&gt;</a>	Yes	The list of items required for this job to run. Only <a href="#">JobRequirement.features</a> is valid during PUT.
<b>reservationRequested</b>	<a href="#">DomainProxy</a>	Yes	The reservation that the job requested.
<b>resourceFailPolicy</b>	<a href="#">JobResourceFailPolicyType</a>	Yes	The policy that dictates what should happen to the job if it is running and at least one of the resources it is using fails.
<b>resourceManagerExtension</b>	String	Yes	If provided during POST, this string will be added to the resource manager extension section of the job submission. Example: "bandwidth=120;queuejob=false" Note that the delimiter between resourceManagerExtension elements is the semicolon.
<b>resourceManagers</b>	<a href="#">Set&lt;ResourceManager&gt;</a>	No	The list of resource managers associated with this job.
<b>rmStandardErrorFilePath</b>	String	No	The path to the remote file containing the standard error of the job.
<b>rmStandardOutputFilePath</b>	String	No	The path to the remote file containing the standard output of the job.

Field Name	Type	POST	Description
<b>standardErrorFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard output of the job.
<b>startCount</b>	<b>Integer</b>	No	The number of times the job has been started.
<b>states</b>	<a href="#"><u>JobStateInformation</u></a>	No	Information about the state of the job.
<b>submitHost</b>	<b>String</b>	Yes	The host from which the job was submitted.
<b>systemJobAction</b>	<b>String</b>	No	The action the system job will take.
<b>systemJobType</b>	<a href="#"><u>JobSystemJobType</u></a>	No	The type of system job. In the Moab Cloud Suite, this will usually be "vmtracking" or "generic."
<b>targetedJobAction</b>	<a href="#"><u>JobActionType</u></a>	No	The action that this job is performing on another job.
<b>targetedJobName</b>	<b>String</b>	No	The name of the job on which this job is performing the targetedJobAction.
<b>templates</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	Yes	The list of all job templates to be set on this job.
<b>triggers</b>	<b>Set&lt;String&gt;</b>	No	The list of triggers associated with this job.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of variables that this job owns or sets on completion.

Field Name	Type	POST	Description
<b>virtualContainers</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	Yes	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>virtualMachines</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	No	The list of virtual machines that are allocated to this job.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	Yes	The requested Virtual Machine Usage Policy for this job.

JobBlock

Field Name	Type	POST	Description
<b>category</b>	<a href="#"><u>JobBlockCategory</u></a>	No	
<b>message</b>	String	No	
<b>type</b>	<a href="#"><u>JobBlockType</u></a>	No	

JobBlockCategory

Value	Description
<b>depend</b>	
<b>jobBlock</b>	
<b>migrate</b>	

JobBlockType

Value	Description
<b>ActivePolicy</b>	
<b>BadUser</b>	
<b>Dependency</b>	
<b>EState</b>	
<b>FairShare</b>	
<b>Hold</b>	
<b>IdlePolicy</b>	
<b>LocalPolicy</b>	
<b>NoClass</b>	
<b>NoData</b>	
<b>NoResource</b>	
<b>NoTime</b>	
<b>PartitionAccess</b>	
<b>Priority</b>	
<b>RMSubmissionFailure</b>	
<b>StartDate</b>	
<b>State</b>	
<b>SysLimits</b>	

JobCredentials

Moab Workload Manager supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	POST	Description
<b>account</b>	<b>String</b>	Yes	The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes.
<b>group</b>	<b>String</b>	Yes	The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.
<b>jobClass</b>	<b>String</b>	Yes	The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab.
<b>qos</b>	<b>String</b>	No	The quality of service assigned to this job.  The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoS's differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.
<b>qosRequested</b>	<b>String</b>	Yes	The quality of service requested for this job.

Field Name	Type	POST	Description
<b>user</b>	<b>String</b>	Yes	The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

JobDates

Field Name	Type	POST	Description
<b>completedDate</b>	<b>Date</b>	No	
<b>createdDate</b>	<b>Date</b>	No	
<b>deadlineDate</b>	<b>Date</b>	Yes	The deadline for completion of the job.
<b>dispatchedDate</b>	<b>Date</b>	No	
<b>earliestRequestedStartDate</b>	<b>Date</b>	Yes	The job will start no sooner than this date.
<b>earliestStartDate</b>	<b>Date</b>	No	
<b>eligibleDate</b>	<b>Date</b>	No	
<b>lastCanceledDate</b>	<b>Date</b>	No	
<b>lastChargedDate</b>	<b>Date</b>	No	
<b>lastPreemptedDate</b>	<b>Date</b>	No	
<b>lastUpdatedDate</b>	<b>Date</b>	No	
<b>startDate</b>	<b>Date</b>	No	
<b>submitDate</b>	<b>Date</b>	No	
<b>terminationDate</b>	<b>Date</b>	No	

JobDependency

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object on on which the job is dependent.
<b>type</b>	<a href="#"><u>JobDependencyType</u></a>	Yes	The type of job dependency. Only set is valid for POST.
<b>value</b>	<b>String</b>	No	

JobDependencyType

Represents the type of a job dependency. For now, only the "set" type is supported.

Value	Description
<b>set</b>	

JobEmailNotifyType

Value	Description
<b>JobStart</b>	
<b>JobEnd</b>	
<b>JobFail</b>	
<b>All</b>	

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	

Value	Description
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.

Value	Description
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.

Value	Description
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldReason

Value	Description
<b>Admin</b>	
<b>NoResources</b>	
<b>SystemLimitsExceeded</b>	
<b>BankFailure</b>	
<b>CannotDebitAccount</b>	

Value	Description
<b>InvalidAccount</b>	
<b>RMFailure</b>	
<b>RMReject</b>	
<b>PolicyViolation</b>	
<b>CredAccess</b>	
<b>CredHold</b>	
<b>PreReq</b>	
<b>Data</b>	
<b>Security</b>	
<b>MissingDependency</b>	

JobHoldType

Value	Description
<b>User</b>	
<b>System</b>	
<b>Batch</b>	
<b>Defer</b>	
<b>All</b>	

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object.

Message

Field Name	Type	POST	Description
<b>count</b>	<b>Integer</b>	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	The date this message was created.
<b>expireDate</b>	<b>Date</b>	No	The date this message expires.
<b>message</b>	<b>String</b>	No	The message itself.

JobHostListMode

Value	Description
<b>superset</b>	
<b>subset</b>	
<b>exactset</b>	

JobPriority

Field Name	Type	POST	Description
<b>run</b>	<b>Long</b>	No	
<b>start</b>	<b>Long</b>	No	
<b>system</b>	<b>Long</b>	No	

Field Name	Type	POST	Description
<b>user</b>	<b>Long</b>	Yes	The user-requested priority for the job. By default, the range is between -1024 and 0. To enable priority range from -1024 to +1023, set <code>ENABLEPOSUSERPRIORITY</code> in the <code>moab.cfg</code> file.

JobQueueStatus

Value	Description
<b>active</b>	
<b>blocked</b>	
<b>completed</b>	
<b>eligible</b>	

JobRejectPolicy

Value	Description
<b>CANCEL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>MAIL</b>	
<b>RETRY</b>	

JobRequirement

Field Name	Type	POST	Description
<b>architecture</b>	<b>String</b>	Yes	The architecture required by the job.
<b>attributes</b>	<a href="#">Map&lt;String, JobRequirementAttribute&gt;</a>	Yes	Required node attributes with version number support.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	The list of node features the job is scheduled against.
<b>featuresExcluded</b>	<b>Set&lt;String&gt;</b>	Yes	Excluded node features. That is, do not select nodes with these features. (See also: <a href="#">featuresExcludedMode</a> .)
<b>featuresExcludedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether excluded features should be AND'ed or OR'd. The default is AND. Only relevant if featuresExcluded is provided. (See also: <a href="#">featuresExcluded</a> .)
<b>featuresRequested</b>	<b>Set&lt;String&gt;</b>	Yes	Requested node features. (See also: <a href="#">featuresRequestedMode</a> .)
<b>featuresRequestedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether requested features should be AND'ed or OR'd. The default is AND. Only relevant if featuresRequested is provided. (See also: <a href="#">featuresRequested</a> .)
<b>image</b>	<b>String</b>	Yes	The image required by the job.
<b>index</b>	<b>Integer</b>	No	The index of the requirement, starting with 0.
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	No	Generic metrics associated with the job as reported by the resource manager.
<b>nodeAccessPolicy</b>	<a href="#">NodeAccessPolicy</a>	Yes	Specifies how node resources should be accessed. Note: If the job requirements array has more than one element that contains nodeAccessPolicy, only the first occurrence will be used.

Field Name	Type	POST	Description
<b>nodeAllocationPolicy</b>	<a href="#">NodeAllocationPolicy</a>	Yes	Specifies how node resources should be selected and allocated to the job. Note: If the job requirements array has more than one element that contains nodeAllocationPolicy, only the first occurrence will be used.
<b>nodeCount</b>	<b>Integer</b>	Yes	The number of nodes required by the job.
<b>nodeSet</b>	<b>String</b>	Yes	<p>The requested node set of the job. This must follow the format <code>SETSELECTION:SETTYPE[:SETLIST]</code></p> <ul style="list-style-type: none"> <li>• SETSELECTION - ANYOF, ONEOF, or FIRSTOF</li> <li>• SETTYPE - FEATURE or VARATTR</li> <li>• SETLIST - For FEATURE, a comma-separated list of features. For VARATTR, a key=value pair.</li> </ul> <p>Examples:</p> <ul style="list-style-type: none"> <li>• ONEOF:FEATURE:fastos,hiprio,bigmem</li> <li>• FIRSTOF:VARATTR:datacenter=Provo:atacenter=SaltLake</li> </ul>
<b>nodes</b>	<a href="#">Set&lt;AllocatedNode&gt;</a>	No	Nodes that have been allocated to meet this requirement.
<b>reservation</b>	<a href="#">DomainProxy</a>	No	The allocated reservation (assigned after the job has a reservation).
<b>resourcesPerTask</b>	<a href="#">Map&lt;String, JobResource&gt;</a>	Yes	Contains requirements for disk, memory, processors, swap, and generic resources. For disk, memory, and swap, the unit is MB. For each resource, the "dedicated" field can be set during POST.
<b>taskCount</b>	<b>Integer</b>	Yes	The number of tasks (processors) required by this job.
<b>tasksPerNode</b>	<b>Integer</b>	Yes	The number of tasks to map to each node.

JobRequirementAttribute

Field Name	Type	POST	Description
<b>comparator</b>	<b>String</b>	Yes	The comparison operator. Valid values: <ul style="list-style-type: none"> <li>• &gt;= - Greater than or equal to</li> <li>• &gt; - Greater than</li> <li>• &lt;= - Less than</li> <li>• &lt; - Less than</li> <li>• %= - Equals</li> <li>• %! - Not equals</li> <li>• Null - Defaults to %=</li> <li>• = - (Deprecated) Equivalent to %=</li> </ul>
<b>displayValue</b>	<b>String</b>	Yes	The display value for the required attribute.
<b>restriction</b>	<a href="#"><u>JobRequirementAttributeRestriction</u></a>	Yes	The restriction of this attribute. May be null, but defaults to <a href="#"><u>JobRequirementAttributeRestriction.must</u></a> .
<b>value</b>	<b>String</b>	Yes	The value of the required attribute. During POST, if value is missing, blank, or null, do not provide a comparator.

JobRequirementAttributeRestriction

Represents a restriction for a job requirement attribute.

Value	Description
<b>must</b>	

JobRequirementFeaturesMode

Value	Description
<b>OR</b>	
<b>AND</b>	

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

NodeAllocationPolicy

Node Allocation enumeration.

Value	Description
<b>FIRSTSET</b>	

Value	Description
<b>MINGLOBAL</b>	
<b>MINLOCAL</b>	
<b>PLUGIN</b>	
<b>NONE</b>	No node allocation policy is specified. Moab defaults to MINRESOURCE when this is the case.
<b>FIRSTAVAILABLE</b>	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, very fast algorithm.
<b>LASTAVAILABLE</b>	This algorithm selects resources so as to minimize the amount of time after the job and before the trailing reservation. This algorithm is a best fit in time algorithm which minimizes the impact of reservation based node-time fragmentation. It is useful in systems where a large number of reservations (job, standing, or administrative) are in place.
<b>MINRESOURCE</b>	This algorithm prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources which still meet the job's resource constraints are selected.
<b>CPULOAD</b>	Nodes are selected which have the maximum amount of available, unused cpu power, i.e. [# of CPU's] - [CPU load]. Good algorithm for timesharing node systems. This algorithm is only applied to jobs starting immediately. For the purpose of future reservations, the <b>MINRESOURCE</b> algorithm is used.
<b>LOCAL</b>	This will call the locally created <b>contrib</b> node allocation algorithm.
<b>CONTIGUOUS</b>	This algorithm will allocate nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	This algorithm will attempt to allocate the most 'balanced' set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node speed. Thus, if possible, nodes with identical speeds will be allocated to the job. If identical speed nodes cannot be found, the algorithm will allocate the set of nodes with the minimum node speed 'span' or range.

Value	Description
<b>PRIORITY</b>	This algorithm allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity.
<b>FASTEST</b>	This algorithm will select nodes in 'fastest node first' order. Nodes will be selected by node speed if specified. If node speed is not specified, nodes will be selected by processor speed. If neither is specified, nodes will be selected in a random order.
<b>PROCESSORLOAD</b>	Alias for CPULOAD.
<b>NODESPEED</b>	Alias for FASTEST.
<b>INREPORTEDORDER</b>	Alias for FIRSTAVAILABLE.
<b>INREVERSEREPORTEDORDER</b>	Alias for LASTAVAILABLE.
<b>CUSTOMPRIORITY</b>	Alias for PRIORITY.
<b>PROCESSORSPEEDBALANCE</b>	Alias for MAXBALANCE.
<b>MINIMUMCONFIGUREDRESOURCES</b>	Alias for MINRESOURCE.

#### AllocatedNode

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	No	
<b>taskCount</b>	<b>Integer</b>	No	

#### JobResource

Represents counts of dedicated and utilized resources.

Field Name	Type	POST	Description
<b>dedicated</b>	<b>Integer</b>	No	The amount of this resource that has been allocated for running workload.
<b>utilized</b>	<b>Integer</b>	No	The amount of this resource that is currently reported as utilized by resource managers.

JobResourceFailPolicyType

Value	Description
<b>CANCEL</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>NOTIFY</b>	
<b>REQUEUE</b>	

ResourceManager

Field Name	Type	POST	Description
<b>isDestination</b>	<b>Boolean</b>	No	
<b>isSource</b>	<b>Boolean</b>	No	
<b>jobName</b>	<b>String</b>	No	
<b>name</b>	<b>String</b>	No	

JobStateInformation

Field Name	Type	POST	Description
<b>state</b>	<a href="#">JobState</a>	No	
<b>stateExpected</b>	<a href="#">JobState</a>	No	
<b>stateLastUpdatedDate</b>	Date	No	
<b>subState</b>	<a href="#">JobSubState</a>	No	

JobState

Value	Description
<b>Idle</b>	
<b>Starting</b>	
<b>Running</b>	
<b>Removed</b>	
<b>Completed</b>	
<b>Hold</b>	
<b>Deferred</b>	
<b>Vacated</b>	
<b>NotQueued</b>	
<b>Unknown</b>	
<b>Staging</b>	
<b>Suspended</b>	
<b>Blocked</b>	

JobSubState

Value	Description
<b>Epilogue</b>	
<b>Migrated</b>	
<b>Preempted</b>	
<b>Prologue</b>	

JobSystemJobType

Value	Description
<b>generic</b>	
<b>osprovision</b>	
<b>osprovision2</b>	
<b>poweroff</b>	
<b>poweron</b>	
<b>reset</b>	
<b>storage</b>	
<b>vmmap</b>	
<b>vmmigrate</b>	
<b>vmtracking</b>	

JobActionType

Value	Description
<b>DESTROY</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

## API version 2

JobArray

Job arrays are an easy way to submit many sub-jobs that perform the same work using the same script, but operate on different sets of data. Sub-jobs are the jobs created by an array job and are identified by the array job ID and an index; for example, if 235[1] is an identifier, the number 235 is a job array ID, and 1 is the sub-job.

Field Name	Type	POST	Description
<b>cancellationPolicy</b>	<a href="#">CancellationPolicyInformation</a>	Yes	Represents the cancellation policy to use for the job array.
<b>indexRanges</b>	<a href="#">List&lt;JobArrayIndexRange&gt;</a>	Yes	The index ranges used to generate the sub-job indices. To use hard-coded values, see <a href="#">indexValues</a> .
<b>indexValues</b>	<b>List&lt;Long&gt;</b>	Yes	The index values to use for the sub-jobs. To use ranges, see <a href="#">indexRanges</a> .
<b>jobPrototype</b>	<a href="#">Job</a>	Yes	The definition of the job to use for each sub-job.
<b>name</b>	<b>String</b>	Yes	The name of the job array. In MWS API version 1, this is stored in the <code>name</code> field of the created jobs. In MWS API version 2, this is stored in the <code>customName</code> field of the created jobs.
<b>slotLimit</b>	<b>Long</b>	Yes	(Optional) The number of sub-jobs in the array that can run at a time.

CancellationPolicyInformation

Job arrays can be canceled based on the success or failure of the first or any sub-job. This class represents the failure policies.

Field Name	Type	POST	Description
<b>anyJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of any sub-job. May be used in combination with <a href="#">firstJob</a> .
<b>firstJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of the first sub-job (array index 1). May be used in combination with <a href="#">anyJob</a> .

CancellationPolicy

This enumeration represents job array cancellation policies, and is to be used in combination with CancellationPolicyInformation.

Value	Description
<b>SUCCESS</b>	Cancels the job array if the specified sub-job succeeds.
<b>FAILURE</b>	Cancels the job array if the specified sub-job fails.

JobArrayIndexRange

Represents information about a job index expression. This is used when creating job arrays only.

Field Name	Type	POST	Description
<b>endIndex</b>	<b>Long</b>	Yes	The end of the index range. i.e. 10 for 1-10.
<b>increment</b>	<b>Long</b>	Yes	The increment of the index range, defaults to 1 and must be greater than 0. For a range of 1-10 with an increment of 2, the list of indices will be [1, 3, 5, 7, 9].
<b>startIndex</b>	<b>Long</b>	Yes	The start of the index range. i.e. 1 for 1-10.

Job

This class represents a job in the Moab Workload Manager. A job is a request for compute resources (CPUs, memory, storage) with which the requester can do work for a given amount of time. In an HPC environment, this might be a batch script to perform a Monte Carlo simulation. In a cloud environment, this would be a virtual machine and its associated storage. Moab will evaluate the request and assign the requested resources to the requester based on policies, current demand, and other factors in the data center. A job will also usually have some process that Moab starts automatically at the assigned start time. In an HPC environment, this can be starting a batch script on the assigned nodes. In a cloud environment, this can be starting provisioning processes to create the virtual machine and storage and install software on it.

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier of this job. Note: this field is not user-assigned and is generated by the database.

Field Name	Type	POST	Description
<b>arrayIndex</b>	<b>Long</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>attributes</b>	<b>Set&lt;String&gt;</b>	Yes	The list of generic attributes associated with this job.
<b>blocks</b>	<a href="#">Set&lt;JobBlock&gt;</a>	No	Reasons the job is blocked from running.
<b>bypassCount</b>	<b>Integer</b>	No	The number of times the job has been backfilled.
<b>cancelCount</b>	<b>Integer</b>	No	The number of times a job has received a cancel request.
<b>commandFile</b>	<b>String</b>	Yes	The path to the file that is executed when this job runs. This is the script that will call all the work of this job.
<b>commandLineArguments</b>	<b>String</b>	Yes	The command line arguments passed to the <code>commandFile</code> .
<b>completionCode</b>	<b>Integer</b>	No	The exit code from this job.
<b>cpuTime</b>	<b>Long</b>	No	CPU usage time in seconds as reported by the resource manager.

Field Name	Type	POST	Description
<b>credentials</b>	<a href="#">JobCredentials</a>	Yes	The credentials (user and group, for example) associated with this job.
<b>customName</b>	<b>String</b>	Yes	The user-specified name of this job.
<b>dates</b>	<a href="#">JobDates</a>	Yes	Various dates associated with this job.
<b>deferCount</b>	<b>Integer</b>	No	The number of times a job has been deferred.
<b>dependencies</b>	<a href="#">Set&lt;JobDependency&gt;</a>	Yes	Dependencies that must be fulfilled before the job can start.
<b>description</b>	<b>String</b>	No	The description of the job. Can be set only in a job template.
<b>duration</b>	<b>Long</b>	Yes	The length of time in seconds requested for the job. Note that it is possible to set duration to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>durationActive</b>	<b>Long</b>	No	The length of time in seconds the job has been active or running.
<b>durationQueued</b>	<b>Long</b>	No	The length of time in seconds the job has been eligible to run in the queue.
<b>durationRemaining</b>	<b>Long</b>	No	An estimate of the time remaining, in seconds, before the job will complete.
<b>durationSuspended</b>	<b>Long</b>	No	The length of time in seconds the job has been suspended.

Field Name	Type	POST	Description
<b>emailNotifyAddresses</b>	<b>Set&lt;String&gt;</b>	Yes	The list of addresses to whom email is sent by the execution server.
<b>emailNotifyTypes</b>	<a href="#"><u>Set&lt;JobEmailNotifyType&gt;</u></a>	Yes	The list of email notify types attached to the job.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	Setting this field to true tells the Moab Workload Manager to set various variables, if populated, in the job's environment.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of environment variables for this job.
<b>epilogScript</b>	<b>String</b>	Yes	The path to the TORQUE epilog script.
<b>flags</b>	<a href="#"><u>Set&lt;JobFlag&gt;</u></a>	Yes	The flags that are set on this job.
<b>holdDate</b>	<b>Date</b>	No	The date the most recent hold was placed on the job.
<b>holdReason</b>	<a href="#"><u>JobHoldReason</u></a>	No	The reason the job is on hold.
<b>holds</b>	<a href="#"><u>Set&lt;JobHoldType&gt;</u></a>	Yes	The holds that are set on the job. The "User" hold type is valid during POST.
<b>initialWorkingDirectory</b>	<b>String</b>	Yes	The path to the directory in which the job will be started.
<b>isActive</b>	<b>Boolean</b>	No	True if the job is active, false if the job is complete.
<b>jobGroup</b>	<b>String</b>	Yes	The job group to which this job belongs (different from credentials.group).

Field Name	Type	POST	Description
<b>masterNode</b>	<a href="#">DomainProxy</a>	No	The first node in the list of allocated nodes for this job. For TORQUE jobs, this represents the "mother superior."
<b>memorySecondsDedicated</b>	Double	No	The memory seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>memorySecondsUtilized</b>	Double	No	The memory seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	No	The list of messages associated with the job. The "message" field is valid during PUT.
<b>migrateCount</b>	Integer	No	The number of times the job has been migrated.
<b>minimumPreemptTime</b>	Long	No	The minimum length of time, in seconds, an active job must be running before it is eligible for preemption.
<b>mwmName</b>	String	No	The name of the Moab Workload Manager instance that owns this job.
<b>name</b>	String	No	The name of this job. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>nodesExcluded</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The list of nodes that should not be considered for this job.

Field Name	Type	POST	Description
<b>nodesRequested</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The exact set, superset, or subset of nodes on which this job must run. (See also: <a href="#">nodesRequestedPolicy</a> .)
<b>nodesRequestedPolicy</b>	<a href="#">JobHostListMode</a>	Yes	Indicates an exact set, superset, or subset of nodes on which the job must run. Only relevant if nodesRequested is provided. (See also: <a href="#">nodesRequested</a> .)
<b>partitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	The list of partitions that this job can access.
<b>partitionAccessListRequested</b>	<b>Set&lt;String&gt;</b>	Yes	The list of partitions that this job has requested.
<b>preemptCount</b>	<b>Integer</b>	No	The number of times the job has been preempted.
<b>priorities</b>	<a href="#">JobPriority</a>	Yes	The list of priorities for the job.
<b>processorSecondsDedicated</b>	<b>Double</b>	No	The processor seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>processorSecondsLimit</b>	<b>Double</b>	No	The limit for processorSecondsUtilized.
<b>processorSecondsUtilized</b>	<b>Double</b>	No	The processor seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>prologScript</b>	<b>String</b>	Yes	The path to the TORQUE prolog script.

Field Name	Type	POST	Description
<b>queueStatus</b>	<a href="#">JobQueueStatus</a>	No	The status of the job in its queue.
<b>rejectPolicies</b>	<a href="#">Set&lt;JobRejectPolicy&gt;</a>	No	The list of policies enabled when a job is rejected.
<b>requirements</b>	<a href="#">Set&lt;JobRequirement&gt;</a>	Yes	The list of items required for this job to run. Only <a href="#">JobRequirement.features</a> is valid during PUT.
<b>reservationRequested</b>	<a href="#">DomainProxy</a>	Yes	The reservation that the job requested.
<b>resourceFailPolicy</b>	<a href="#">JobResourceFailPolicyType</a>	Yes	The policy that dictates what should happen to the job if it is running and at least one of the resources it is using fails.
<b>resourceManagerExtension</b>	String	Yes	If provided during POST, this string will be added to the resource manager extension section of the job submission. Example: "bandwidth=120;queuejob=false" Note that the delimiter between resourceManagerExtension elements is the semicolon.
<b>resourceManagers</b>	<a href="#">Set&lt;ResourceManager&gt;</a>	No	The list of resource managers associated with this job.
<b>rmStandardErrorFilePath</b>	String	No	The path to the remote file containing the standard error of the job.
<b>rmStandardOutputFilePath</b>	String	No	The path to the remote file containing the standard output of the job.

Field Name	Type	POST	Description
<b>standardErrorFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard output of the job.
<b>startCount</b>	<b>Integer</b>	No	The number of times the job has been started.
<b>states</b>	<a href="#">JobStateInformation</a>	No	Information about the state of the job.
<b>submitHost</b>	<b>String</b>	Yes	The host from which the job was submitted.
<b>systemJobAction</b>	<b>String</b>	No	The action the system job will take.
<b>systemJobType</b>	<a href="#">JobSystemJobType</a>	No	The type of system job. In the Moab Cloud Suite, this will usually be "vmtracking" or "generic."
<b>targetedJobAction</b>	<a href="#">JobActionType</a>	No	The action that this job is performing on another job.
<b>targetedJobName</b>	<b>String</b>	No	The name of the job on which this job is performing the targetedJobAction.
<b>templates</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The list of all job templates to be set on this job.
<b>triggers</b>	<b>Set&lt;String&gt;</b>	No	The list of triggers associated with this job.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of variables that this job owns or sets on completion.

Field Name	Type	POST	Description
<b>virtualContainers</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	Yes	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>virtualMachines</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	No	The list of virtual machines that are allocated to this job.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	Yes	The requested Virtual Machine Usage Policy for this job.

JobBlock

Field Name	Type	POST	Description
<b>category</b>	<a href="#"><u>JobBlockCategory</u></a>	No	
<b>message</b>	String	No	
<b>type</b>	<a href="#"><u>JobBlockType</u></a>	No	

JobBlockCategory

Value	Description
<b>depend</b>	
<b>jobBlock</b>	
<b>migrate</b>	

JobBlockType

Value	Description
<b>ActivePolicy</b>	
<b>BadUser</b>	
<b>Dependency</b>	
<b>EState</b>	
<b>FairShare</b>	
<b>Hold</b>	
<b>IdlePolicy</b>	
<b>LocalPolicy</b>	
<b>NoClass</b>	
<b>NoData</b>	
<b>NoResource</b>	
<b>NoTime</b>	
<b>PartitionAccess</b>	
<b>Priority</b>	
<b>RMSubmissionFailure</b>	
<b>StartDate</b>	
<b>State</b>	
<b>SysLimits</b>	

JobCredentials

Moab Workload Manager supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	POST	Description
<b>account</b>	<b>String</b>	Yes	The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes.
<b>group</b>	<b>String</b>	Yes	The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.
<b>jobClass</b>	<b>String</b>	Yes	The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab.
<b>qos</b>	<b>String</b>	No	The quality of service assigned to this job.  The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoS's differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.
<b>qosRequested</b>	<b>String</b>	Yes	The quality of service requested for this job.

Field Name	Type	POST	Description
<b>user</b>	<b>String</b>	Yes	The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

JobDates

Field Name	Type	POST	Description
<b>completedDate</b>	<b>Date</b>	No	
<b>createdDate</b>	<b>Date</b>	No	
<b>deadlineDate</b>	<b>Date</b>	Yes	The deadline for completion of the job.
<b>dispatchedDate</b>	<b>Date</b>	No	
<b>earliestRequestedStartDate</b>	<b>Date</b>	Yes	The job will start no sooner than this date.
<b>earliestStartDate</b>	<b>Date</b>	No	
<b>eligibleDate</b>	<b>Date</b>	No	
<b>lastCanceledDate</b>	<b>Date</b>	No	
<b>lastChargedDate</b>	<b>Date</b>	No	
<b>lastPreemptedDate</b>	<b>Date</b>	No	
<b>lastUpdatedDate</b>	<b>Date</b>	No	
<b>startDate</b>	<b>Date</b>	No	
<b>submitDate</b>	<b>Date</b>	No	
<b>terminationDate</b>	<b>Date</b>	No	

JobDependency

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object on on which the job is dependent.
<b>type</b>	<a href="#"><u>JobDependencyType</u></a>	Yes	The type of job dependency. Only set is valid for POST.
<b>value</b>	<b>String</b>	No	

JobDependencyType

Represents the type of a job dependency. For now, only the "set" type is supported.

Value	Description
<b>set</b>	

JobEmailNotifyType

Value	Description
<b>JobStart</b>	
<b>JobEnd</b>	
<b>JobFail</b>	
<b>All</b>	

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	

Value	Description
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.

Value	Description
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.

Value	Description
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldReason

Value	Description
<b>Admin</b>	
<b>NoResources</b>	
<b>SystemLimitsExceeded</b>	
<b>BankFailure</b>	
<b>CannotDebitAccount</b>	

Value	Description
<b>InvalidAccount</b>	
<b>RMFailure</b>	
<b>RMReject</b>	
<b>PolicyViolation</b>	
<b>CredAccess</b>	
<b>CredHold</b>	
<b>PreReq</b>	
<b>Data</b>	
<b>Security</b>	
<b>MissingDependency</b>	

#### JobHoldType

Value	Description
<b>User</b>	
<b>System</b>	
<b>Batch</b>	
<b>Defer</b>	
<b>All</b>	

#### DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object.

Message

Field Name	Type	POST	Description
<b>count</b>	<b>Integer</b>	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	The date this message was created.
<b>expireDate</b>	<b>Date</b>	No	The date this message expires.
<b>message</b>	<b>String</b>	No	The message itself.

JobHostListMode

Value	Description
<b>superset</b>	
<b>subset</b>	
<b>exactset</b>	

JobPriority

Field Name	Type	POST	Description
<b>run</b>	<b>Long</b>	No	
<b>start</b>	<b>Long</b>	No	
<b>system</b>	<b>Long</b>	No	

Field Name	Type	POST	Description
<b>user</b>	<b>Long</b>	Yes	The user-requested priority for the job. By default, the range is between -1024 and 0. To enable priority range from -1024 to +1023, set <code>ENABLEPOSUSERPRIORITY</code> in the <code>moab.cfg</code> file.

JobQueueStatus

Value	Description
<b>active</b>	
<b>blocked</b>	
<b>completed</b>	
<b>eligible</b>	

JobRejectPolicy

Value	Description
<b>CANCEL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>MAIL</b>	
<b>RETRY</b>	

JobRequirement

Field Name	Type	POST	Description
<b>architecture</b>	<b>String</b>	Yes	The architecture required by the job.
<b>attributes</b>	<a href="#">Map&lt;String, JobRequirementAttribute&gt;</a>	Yes	Required node attributes with version number support.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	The list of node features the job is scheduled against.
<b>featuresExcluded</b>	<b>Set&lt;String&gt;</b>	Yes	Excluded node features. That is, do not select nodes with these features. (See also: <a href="#">featuresExcludedMode</a> .)
<b>featuresExcludedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether excluded features should be AND'ed or OR'd. The default is AND. Only relevant if featuresExcluded is provided. (See also: <a href="#">featuresExcluded</a> .)
<b>featuresRequested</b>	<b>Set&lt;String&gt;</b>	Yes	Requested node features. (See also: <a href="#">featuresRequestedMode</a> .)
<b>featuresRequestedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether requested features should be AND'ed or OR'd. The default is AND. Only relevant if featuresRequested is provided. (See also: <a href="#">featuresRequested</a> .)
<b>image</b>	<b>String</b>	Yes	The image required by the job.
<b>index</b>	<b>Integer</b>	No	The index of the requirement, starting with 0.
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	No	Generic metrics associated with the job as reported by the resource manager.
<b>nodeAccessPolicy</b>	<a href="#">NodeAccessPolicy</a>	Yes	Specifies how node resources should be accessed. Note: If the job requirements array has more than one element that contains nodeAccessPolicy, only the first occurrence will be used.

Field Name	Type	POST	Description
<b>nodeAllocationPolicy</b>	<a href="#">NodeAllocationPolicy</a>	Yes	Specifies how node resources should be selected and allocated to the job. Note: If the job requirements array has more than one element that contains nodeAllocationPolicy, only the first occurrence will be used.
<b>nodeCount</b>	<b>Integer</b>	Yes	The number of nodes required by the job.
<b>nodeSet</b>	<b>String</b>	Yes	<p>The requested node set of the job. This must follow the format <code>SETSELECTION:SETTYPE[:SETLIST]</code></p> <ul style="list-style-type: none"> <li>• SETSELECTION - ANYOF, ONEOF, or FIRSTOF</li> <li>• SETTYPE - FEATURE or VARATTR</li> <li>• SETLIST - For FEATURE, a comma-separated list of features. For VARATTR, a key=value pair.</li> </ul> <p>Examples:</p> <ul style="list-style-type: none"> <li>• ONEOF:FEATURE:fastos,hiprio,bigmem</li> <li>• FIRSTOF:VARATTR:datacenter=Provo:atacenter=SaltLake</li> </ul>
<b>nodes</b>	<a href="#">Set&lt;AllocatedNode&gt;</a>	No	Nodes that have been allocated to meet this requirement.
<b>reservation</b>	<a href="#">DomainProxy</a>	No	The allocated reservation (assigned after the job has a reservation).
<b>resourcesPerTask</b>	<a href="#">Map&lt;String, JobResource&gt;</a>	Yes	Contains requirements for disk, memory, processors, swap, and generic resources. For disk, memory, and swap, the unit is MB. For each resource, the "dedicated" field can be set during POST.
<b>taskCount</b>	<b>Integer</b>	Yes	The number of tasks (processors) required by this job.
<b>tasksPerNode</b>	<b>Integer</b>	Yes	The number of tasks to map to each node.

JobRequirementAttribute

Field Name	Type	POST	Description
<b>comparator</b>	<b>String</b>	Yes	The comparison operator. Valid values: <ul style="list-style-type: none"> <li>• &gt;= - Greater than or equal to</li> <li>• &gt; - Greater than</li> <li>• &lt;= - Less than</li> <li>• &lt; - Less than</li> <li>• %= - Equals</li> <li>• %! - Not equals</li> <li>• Null - Defaults to %=</li> <li>• = - (Deprecated) Equivalent to %=</li> </ul>
<b>displayValue</b>	<b>String</b>	Yes	The display value for the required attribute.
<b>restriction</b>	<a href="#"><u>JobRequirementAttributeRestriction</u></a>	Yes	The restriction of this attribute. May be null, but defaults to <a href="#"><u>JobRequirementAttributeRestriction.must</u></a> .
<b>value</b>	<b>String</b>	Yes	The value of the required attribute. During POST, if value is missing, blank, or null, do not provide a comparator.

JobRequirementAttributeRestriction

Represents a restriction for a job requirement attribute.

Value	Description
<b>must</b>	

JobRequirementFeaturesMode

Value	Description
<b>OR</b>	
<b>AND</b>	

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

NodeAllocationPolicy

Node Allocation enumeration.

Value	Description
<b>FIRSTSET</b>	

Value	Description
<b>MINGLOBAL</b>	
<b>MINLOCAL</b>	
<b>PLUGIN</b>	
<b>NONE</b>	No node allocation policy is specified. Moab defaults to MINRESOURCE when this is the case.
<b>FIRSTAVAILABLE</b>	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, very fast algorithm.
<b>LASTAVAILABLE</b>	This algorithm selects resources so as to minimize the amount of time after the job and before the trailing reservation. This algorithm is a best fit in time algorithm which minimizes the impact of reservation based node-time fragmentation. It is useful in systems where a large number of reservations (job, standing, or administrative) are in place.
<b>MINRESOURCE</b>	This algorithm prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources which still meet the job's resource constraints are selected.
<b>CPULOAD</b>	Nodes are selected which have the maximum amount of available, unused cpu power, i.e. [# of CPU's] - [CPU load]. Good algorithm for timesharing node systems. This algorithm is only applied to jobs starting immediately. For the purpose of future reservations, the <b>MINRESOURCE</b> algorithm is used.
<b>LOCAL</b>	This will call the locally created <b>contrib</b> node allocation algorithm.
<b>CONTIGUOUS</b>	This algorithm will allocate nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	This algorithm will attempt to allocate the most 'balanced' set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node speed. Thus, if possible, nodes with identical speeds will be allocated to the job. If identical speed nodes cannot be found, the algorithm will allocate the set of nodes with the minimum node speed 'span' or range.

Value	Description
<b>PRIORITY</b>	This algorithm allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity.
<b>FASTEST</b>	This algorithm will select nodes in 'fastest node first' order. Nodes will be selected by node speed if specified. If node speed is not specified, nodes will be selected by processor speed. If neither is specified, nodes will be selected in a random order.
<b>PROCESSORLOAD</b>	Alias for CPULOAD.
<b>NODESPEED</b>	Alias for FASTEST.
<b>INREPORTEDORDER</b>	Alias for FIRSTAVAILABLE.
<b>INREVERSEREPORTEDORDER</b>	Alias for LASTAVAILABLE.
<b>CUSTOMPRIORITY</b>	Alias for PRIORITY.
<b>PROCESSORSPEEDBALANCE</b>	Alias for MAXBALANCE.
<b>MINIMUMCONFIGUREDRESOURCES</b>	Alias for MINRESOURCE.

AllocatedNode

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	No	
<b>taskCount</b>	<b>Integer</b>	No	

JobResource

Represents counts of dedicated and utilized resources.

Field Name	Type	POST	Description
<b>dedicated</b>	<b>Integer</b>	No	The amount of this resource that has been allocated for running workload.
<b>utilized</b>	<b>Integer</b>	No	The amount of this resource that is currently reported as utilized by resource managers.

JobResourceFailPolicyType

Value	Description
<b>CANCEL</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>NOTIFY</b>	
<b>REQUEUE</b>	

ResourceManager

Field Name	Type	POST	Description
<b>isDestination</b>	<b>Boolean</b>	No	
<b>isSource</b>	<b>Boolean</b>	No	
<b>jobName</b>	<b>String</b>	No	
<b>name</b>	<b>String</b>	No	

JobStateInformation

Field Name	Type	POST	Description
<b>state</b>	<a href="#">JobState</a>	No	
<b>stateExpected</b>	<a href="#">JobState</a>	No	
<b>stateLastUpdatedDate</b>	Date	No	
<b>subState</b>	<a href="#">JobSubState</a>	No	

JobState

Value	Description
<b>Idle</b>	
<b>Starting</b>	
<b>Running</b>	
<b>Removed</b>	
<b>Completed</b>	
<b>Hold</b>	
<b>Deferred</b>	
<b>Vacated</b>	
<b>NotQueued</b>	
<b>Unknown</b>	
<b>Staging</b>	
<b>Suspended</b>	
<b>Blocked</b>	

JobSubState

Value	Description
<b>Epilogue</b>	
<b>Migrated</b>	
<b>Preempted</b>	
<b>Prologue</b>	

JobSystemJobType

Value	Description
<b>generic</b>	
<b>osprovision</b>	
<b>osprovision2</b>	
<b>poweroff</b>	
<b>poweron</b>	
<b>reset</b>	
<b>storage</b>	
<b>vmmap</b>	
<b>vmmigrate</b>	
<b>vmtracking</b>	

JobActionType

Value	Description
<b>DESTROY</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

## API version 1

JobArray

Job arrays are an easy way to submit many sub-jobs that perform the same work using the same script, but operate on different sets of data. Sub-jobs are the jobs created by an array job and are identified by the array job ID and an index; for example, if 235[1] is an identifier, the number 235 is a job array ID, and 1 is the sub-job.

Field Name	Type	POST	Description
<b>cancellationPolicy</b>	<a href="#">CancellationPolicyInformation</a>	Yes	Represents the cancellation policy to use for the job array.
<b>indexRanges</b>	<a href="#">List&lt;JobArrayIndexRange&gt;</a>	Yes	The index ranges used to generate the sub-job indices. To use hard-coded values, see <a href="#">indexValues</a> .
<b>indexValues</b>	<b>List&lt;Long&gt;</b>	Yes	The index values to use for the sub-jobs. To use ranges, see <a href="#">indexRanges</a> .
<b>jobPrototype</b>	<a href="#">Job</a>	Yes	The definition of the job to use for each sub-job.
<b>name</b>	<b>String</b>	Yes	The name of the job array. In MWS API version 1, this is stored in the <code>name</code> field of the created jobs. In MWS API version 2, this is stored in the <code>customName</code> field of the created jobs.
<b>slotLimit</b>	<b>Long</b>	Yes	(Optional) The number of sub-jobs in the array that can run at a time.

CancellationPolicyInformation

Job arrays can be canceled based on the success or failure of the first or any sub-job. This class represents the failure policies.

Field Name	Type	POST	Description
<b>anyJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of any sub-job. May be used in combination with <a href="#">firstJob</a> .
<b>firstJob</b>	<a href="#">CancellationPolicy</a>	Yes	The cancellation policy based on the result of the first sub-job (array index 1). May be used in combination with <a href="#">anyJob</a> .

CancellationPolicy

This enumeration represents job array cancellation policies, and is to be used in combination with CancellationPolicyInformation.

Value	Description
<b>SUCCESS</b>	Cancels the job array if the specified sub-job succeeds.
<b>FAILURE</b>	Cancels the job array if the specified sub-job fails.

JobArrayIndexRange

Represents information about a job index expression. This is used when creating job arrays only.

Field Name	Type	POST	Description
<b>endIndex</b>	<b>Long</b>	Yes	The end of the index range. i.e. 10 for 1-10.
<b>increment</b>	<b>Long</b>	Yes	The increment of the index range, defaults to 1 and must be greater than 0. For a range of 1-10 with an increment of 2, the list of indices will be [1, 3, 5, 7, 9].
<b>startIndex</b>	<b>Long</b>	Yes	The start of the index range. i.e. 1 for 1-10.

Job

This class represents a job in the Moab Workload Manager. A job is a request for compute resources (CPUs, memory, storage) with which the requester can do work for a given amount of time. In an HPC environment, this might be a batch script to perform a Monte Carlo simulation. In a cloud environment, this would be a virtual machine and its associated storage. Moab will evaluate the request and assign the requested resources to the requester based on policies, current demand, and other factors in the data center. A job will also usually have some process that Moab starts automatically at the assigned start time. In an HPC environment, this can be starting a batch script on the assigned nodes. In a cloud environment, this can be starting provisioning processes to create the virtual machine and storage and install software on it.

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier of this job. Note: this field is not user-assigned and is generated by the database.

Field Name	Type	POST	Description
<b>arrayIndex</b>	<b>Long</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>attributes</b>	<b>Set&lt;String&gt;</b>	Yes	The list of generic attributes associated with this job.
<b>blocks</b>	<b><a href="#">Set&lt;JobBlock&gt;</a></b>	No	Reasons the job is blocked from running.
<b>bypassCount</b>	<b>Integer</b>	No	The number of times the job has been backfilled.
<b>cancelCount</b>	<b>Integer</b>	No	The number of times a job has received a cancel request.
<b>commandFile</b>	<b>String</b>	Yes	The path to the file that is executed when this job runs. This is the script that will call all the work of this job.
<b>commandLineArguments</b>	<b>String</b>	Yes	The command line arguments passed to the <code>commandFile</code> .
<b>completionCode</b>	<b>Integer</b>	No	The exit code from this job.
<b>cpuTime</b>	<b>Long</b>	No	CPU usage time in seconds as reported by the resource manager.

Field Name	Type	POST	Description
<b>credentials</b>	<a href="#">JobCredentials</a>	Yes	The credentials (user and group, for example) associated with this job.
<b>customName</b>	<b>String</b>	Yes	The user-specified name of this job.
<b>dates</b>	<a href="#">JobDates</a>	Yes	Various dates associated with this job.
<b>deferCount</b>	<b>Integer</b>	No	The number of times a job has been deferred.
<b>dependencies</b>	<a href="#">Set&lt;JobDependency&gt;</a>	Yes	Dependencies that must be fulfilled before the job can start.
<b>description</b>	<b>String</b>	No	The description of the job. Can be set only in a job template.
<b>duration</b>	<b>Long</b>	Yes	The length of time in seconds requested for the job. Note that it is possible to set duration to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>durationActive</b>	<b>Long</b>	No	The length of time in seconds the job has been active or running.
<b>durationQueued</b>	<b>Long</b>	No	The length of time in seconds the job has been eligible to run in the queue.
<b>durationRemaining</b>	<b>Long</b>	No	An estimate of the time remaining, in seconds, before the job will complete.
<b>durationSuspended</b>	<b>Long</b>	No	The length of time in seconds the job has been suspended.

Field Name	Type	POST	Description
<b>emailNotifyAddresses</b>	<b>Set&lt;String&gt;</b>	Yes	The list of addresses to whom email is sent by the execution server.
<b>emailNotifyTypes</b>	<a href="#"><u>Set&lt;JobEmailNotifyType&gt;</u></a>	Yes	The list of email notify types attached to the job.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	Setting this field to true tells the Moab Workload Manager to set various variables, if populated, in the job's environment.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of environment variables for this job.
<b>epilogScript</b>	<b>String</b>	Yes	The path to the TORQUE epilog script.
<b>flags</b>	<a href="#"><u>Set&lt;JobFlag&gt;</u></a>	Yes	The flags that are set on this job.
<b>holdDate</b>	<b>Date</b>	No	The date the most recent hold was placed on the job.
<b>holdReason</b>	<a href="#"><u>JobHoldReason</u></a>	No	The reason the job is on hold.
<b>holds</b>	<a href="#"><u>Set&lt;JobHoldType&gt;</u></a>	Yes	The holds that are set on the job. The "User" hold type is valid during POST.
<b>initialWorkingDirectory</b>	<b>String</b>	Yes	The path to the directory in which the job will be started.
<b>isActive</b>	<b>Boolean</b>	No	True if the job is active, false if the job is complete.
<b>jobGroup</b>	<b>String</b>	Yes	The job group to which this job belongs (different from credentials.group).

Field Name	Type	POST	Description
<b>masterNode</b>	<a href="#">DomainProxy</a>	No	The first node in the list of allocated nodes for this job. For TORQUE jobs, this represents the "mother superior."
<b>memorySecondsDedicated</b>	Double	No	The memory seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>memorySecondsUtilized</b>	Double	No	The memory seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	No	The list of messages associated with the job. The "message" field is valid during PUT.
<b>migrateCount</b>	Integer	No	The number of times the job has been migrated.
<b>minimumPreemptTime</b>	Long	No	The minimum length of time, in seconds, an active job must be running before it is eligible for preemption.
<b>mwmName</b>	String	No	The name of the Moab Workload Manager instance that owns this job.
<b>name</b>	String	No	The name of this job. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>nodesExcluded</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The list of nodes that should not be considered for this job.

Field Name	Type	POST	Description
<b>nodesRequested</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	The exact set, superset, or subset of nodes on which this job must run. (See also: <a href="#">nodesRequestedPolicy</a> .)
<b>nodesRequestedPolicy</b>	<a href="#">JobHostListMode</a>	Yes	Indicates an exact set, superset, or subset of nodes on which the job must run. Only relevant if nodesRequested is provided. (See also: <a href="#">nodesRequested</a> .)
<b>partitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	The list of partitions that this job can access.
<b>partitionAccessListRequested</b>	<b>Set&lt;String&gt;</b>	Yes	The list of partitions that this job has requested.
<b>preemptCount</b>	<b>Integer</b>	No	The number of times the job has been preempted.
<b>priorities</b>	<a href="#">JobPriority</a>	Yes	The list of priorities for the job.
<b>processorSecondsDedicated</b>	<b>Double</b>	No	The processor seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>processorSecondsLimit</b>	<b>Double</b>	No	The limit for processorSecondsUtilized.
<b>processorSecondsUtilized</b>	<b>Double</b>	No	The processor seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>prologScript</b>	<b>String</b>	Yes	The path to the TORQUE prolog script.

Field Name	Type	POST	Description
<b>queueStatus</b>	<a href="#">JobQueueStatus</a>	No	The status of the job in its queue.
<b>rejectPolicies</b>	<a href="#">Set&lt;JobRejectPolicy&gt;</a>	No	The list of policies enabled when a job is rejected.
<b>requirements</b>	<a href="#">Set&lt;JobRequirement&gt;</a>	Yes	The list of items required for this job to run. Only <a href="#">JobRequirement.features</a> is valid during PUT.
<b>reservationRequested</b>	<a href="#">DomainProxy</a>	Yes	The reservation that the job requested.
<b>resourceFailPolicy</b>	<a href="#">JobResourceFailPolicyType</a>	Yes	The policy that dictates what should happen to the job if it is running and at least one of the resources it is using fails.
<b>resourceManagerExtension</b>	String	Yes	If provided during POST, this string will be added to the resource manager extension section of the job submission. Example: "bandwidth=120;queuejob=false" Note that the delimiter between resourceManagerExtension elements is the semicolon.
<b>resourceManagers</b>	<a href="#">Set&lt;ResourceManager&gt;</a>	No	The list of resource managers associated with this job.
<b>rmStandardErrorFilePath</b>	String	No	The path to the remote file containing the standard error of the job.
<b>rmStandardOutputFilePath</b>	String	No	The path to the remote file containing the standard output of the job.

Field Name	Type	POST	Description
<b>standardErrorFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	<b>String</b>	Yes	The path to the file containing the standard output of the job.
<b>startCount</b>	<b>Integer</b>	No	The number of times the job has been started.
<b>states</b>	<a href="#"><u>JobStateInformation</u></a>	No	Information about the state of the job.
<b>submitHost</b>	<b>String</b>	Yes	The host from which the job was submitted.
<b>systemJobAction</b>	<b>String</b>	No	The action the system job will take.
<b>systemJobType</b>	<a href="#"><u>JobSystemJobType</u></a>	No	The type of system job. In the Moab Cloud Suite, this will usually be "vmtracking" or "generic."
<b>targetedJobAction</b>	<a href="#"><u>JobActionType</u></a>	No	The action that this job is performing on another job.
<b>targetedJobName</b>	<b>String</b>	No	The name of the job on which this job is performing the targetedJobAction.
<b>templates</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	Yes	The list of all job templates to be set on this job.
<b>triggers</b>	<b>Set&lt;String&gt;</b>	No	The list of triggers associated with this job.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	The list of variables that this job owns or sets on completion.

Field Name	Type	POST	Description
<b>virtualContainers</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	Yes	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>virtualMachines</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	No	The list of virtual machines that are allocated to this job.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	Yes	The requested Virtual Machine Usage Policy for this job.

JobBlock

Field Name	Type	POST	Description
<b>category</b>	<a href="#"><u>JobBlockCategory</u></a>	No	
<b>message</b>	String	No	
<b>type</b>	<a href="#"><u>JobBlockType</u></a>	No	

JobBlockCategory

Value	Description
<b>depend</b>	
<b>jobBlock</b>	
<b>migrate</b>	

JobBlockType

Value	Description
<b>ActivePolicy</b>	
<b>BadUser</b>	
<b>Dependency</b>	
<b>EState</b>	
<b>FairShare</b>	
<b>Hold</b>	
<b>IdlePolicy</b>	
<b>LocalPolicy</b>	
<b>NoClass</b>	
<b>NoData</b>	
<b>NoResource</b>	
<b>NoTime</b>	
<b>PartitionAccess</b>	
<b>Priority</b>	
<b>RMSubmissionFailure</b>	
<b>StartDate</b>	
<b>State</b>	
<b>SysLimits</b>	

JobCredentials

Moab Workload Manager supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	POST	Description
<b>account</b>	<b>String</b>	Yes	The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes.
<b>group</b>	<b>String</b>	Yes	The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.
<b>jobClass</b>	<b>String</b>	Yes	The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab.
<b>qos</b>	<b>String</b>	No	The quality of service assigned to this job.  The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoS's differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.
<b>qosRequested</b>	<b>String</b>	Yes	The quality of service requested for this job.

Field Name	Type	POST	Description
<b>user</b>	<b>String</b>	Yes	The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

JobDates

Field Name	Type	POST	Description
<b>completedDate</b>	<b>Date</b>	No	
<b>createdDate</b>	<b>Date</b>	No	
<b>deadlineDate</b>	<b>Date</b>	Yes	The deadline for completion of the job.
<b>dispatchedDate</b>	<b>Date</b>	No	
<b>earliestRequestedStartDate</b>	<b>Date</b>	Yes	The job will start no sooner than this date.
<b>earliestStartDate</b>	<b>Date</b>	No	
<b>eligibleDate</b>	<b>Date</b>	No	
<b>lastCanceledDate</b>	<b>Date</b>	No	
<b>lastChargedDate</b>	<b>Date</b>	No	
<b>lastPreemptedDate</b>	<b>Date</b>	No	
<b>lastUpdatedDate</b>	<b>Date</b>	No	
<b>startDate</b>	<b>Date</b>	No	
<b>submitDate</b>	<b>Date</b>	No	
<b>terminationDate</b>	<b>Date</b>	No	

JobDependency

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object on on which the job is dependent.
<b>type</b>	<a href="#"><u>JobDependencyType</u></a>	Yes	The type of job dependency. Only set is valid for POST.
<b>value</b>	<b>String</b>	No	

JobDependencyType

Represents the type of a job dependency. For now, only the "set" type is supported.

Value	Description
<b>set</b>	

JobEmailNotifyType

Value	Description
<b>JobStart</b>	
<b>JobEnd</b>	
<b>JobFail</b>	
<b>All</b>	

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	

Value	Description
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.

Value	Description
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.

Value	Description
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldReason

Value	Description
<b>Admin</b>	
<b>NoResources</b>	
<b>SystemLimitsExceeded</b>	
<b>BankFailure</b>	
<b>CannotDebitAccount</b>	

Value	Description
<b>InvalidAccount</b>	
<b>RMFailure</b>	
<b>RMReject</b>	
<b>PolicyViolation</b>	
<b>CredAccess</b>	
<b>CredHold</b>	
<b>PreReq</b>	
<b>Data</b>	
<b>Security</b>	
<b>MissingDependency</b>	

JobHoldType

Value	Description
<b>User</b>	
<b>System</b>	
<b>Batch</b>	
<b>Defer</b>	
<b>All</b>	

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	Yes	The name of the object.

Message

Field Name	Type	POST	Description
<b>count</b>	<b>Integer</b>	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	The date this message was created.
<b>expireDate</b>	<b>Date</b>	No	The date this message expires.
<b>message</b>	<b>String</b>	No	The message itself.

JobHostListMode

Value	Description
<b>superset</b>	
<b>subset</b>	
<b>exactset</b>	

JobPriority

Field Name	Type	POST	Description
<b>run</b>	<b>Long</b>	No	
<b>start</b>	<b>Long</b>	No	
<b>system</b>	<b>Long</b>	No	

Field Name	Type	POST	Description
<b>user</b>	<b>Long</b>	Yes	The user-requested priority for the job. By default, the range is between -1024 and 0. To enable priority range from -1024 to +1023, set <code>ENABLEPOSUSERPRIORITY</code> in the <code>moab.cfg</code> file.

JobQueueStatus

Value	Description
<b>active</b>	
<b>blocked</b>	
<b>completed</b>	
<b>eligible</b>	

JobRejectPolicy

Value	Description
<b>CANCEL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>MAIL</b>	
<b>RETRY</b>	

JobRequirement

Field Name	Type	POST	Description
<b>architecture</b>	<b>String</b>	Yes	The architecture required by the job.
<b>attributes</b>	<a href="#">Map&lt;String, JobRequirementAttribute&gt;</a>	Yes	Required node attributes with version number support.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	The list of node features the job is scheduled against.
<b>featuresExcluded</b>	<b>Set&lt;String&gt;</b>	Yes	Excluded node features. That is, do not select nodes with these features. (See also: <a href="#">featuresExcludedMode</a> .)
<b>featuresExcludedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether excluded features should be AND'ed or OR'd. The default is AND. Only relevant if featuresExcluded is provided. (See also: <a href="#">featuresExcluded</a> .)
<b>featuresRequested</b>	<b>Set&lt;String&gt;</b>	Yes	Requested node features. (See also: <a href="#">featuresRequestedMode</a> .)
<b>featuresRequestedMode</b>	<a href="#">JobRequirementFeaturesMode</a>	Yes	Indicates whether requested features should be AND'ed or OR'd. The default is AND. Only relevant if featuresRequested is provided. (See also: <a href="#">featuresRequested</a> .)
<b>image</b>	<b>String</b>	Yes	The image required by the job.
<b>index</b>	<b>Integer</b>	No	The index of the requirement, starting with 0.
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	No	Generic metrics associated with the job as reported by the resource manager.
<b>nodeAccessPolicy</b>	<a href="#">NodeAccessPolicy</a>	Yes	Specifies how node resources should be accessed. Note: If the job requirements array has more than one element that contains nodeAccessPolicy, only the first occurrence will be used.

Field Name	Type	POST	Description
<b>nodeAllocationPolicy</b>	<a href="#">NodeAllocationPolicy</a>	Yes	Specifies how node resources should be selected and allocated to the job. Note: If the job requirements array has more than one element that contains nodeAllocationPolicy, only the first occurrence will be used.
<b>nodeCount</b>	<b>Integer</b>	Yes	The number of nodes required by the job.
<b>nodeSet</b>	<b>String</b>	Yes	<p>The requested node set of the job. This must follow the format <code>SETSELECTION:SETTYPE[:SETLIST]</code></p> <ul style="list-style-type: none"> <li>• SETSELECTION - ANYOF, ONEOF, or FIRSTOF</li> <li>• SETTYPE - FEATURE or VARATTR</li> <li>• SETLIST - For FEATURE, a comma-separated list of features. For VARATTR, a key=value pair.</li> </ul> <p>Examples:</p> <ul style="list-style-type: none"> <li>• ONEOF:FEATURE:fastos,hiprio,bigmem</li> <li>• FIRSTOF:VARATTR:datacenter=Provo:atacenter=SaltLake</li> </ul>
<b>nodes</b>	<a href="#">Set&lt;AllocatedNode&gt;</a>	No	Nodes that have been allocated to meet this requirement.
<b>reservation</b>	<a href="#">DomainProxy</a>	No	The allocated reservation (assigned after the job has a reservation).
<b>resourcesPerTask</b>	<a href="#">Map&lt;String, JobResource&gt;</a>	Yes	Contains requirements for disk, memory, processors, swap, and generic resources. For disk, memory, and swap, the unit is MB. For each resource, the "dedicated" field can be set during POST.
<b>taskCount</b>	<b>Integer</b>	Yes	The number of tasks (processors) required by this job.
<b>tasksPerNode</b>	<b>Integer</b>	Yes	The number of tasks to map to each node.

JobRequirementAttribute

Field Name	Type	POST	Description
<b>comparator</b>	<b>String</b>	Yes	The comparison operator. Valid values: <ul style="list-style-type: none"> <li>• &gt;= - Greater than or equal to</li> <li>• &gt; - Greater than</li> <li>• &lt;= - Less than</li> <li>• &lt; - Less than</li> <li>• %= - Equals</li> <li>• %! - Not equals</li> <li>• Null - Defaults to %=</li> <li>• = - (Deprecated) Equivalent to %=</li> </ul>
<b>displayValue</b>	<b>String</b>	Yes	The display value for the required attribute.
<b>restriction</b>	<a href="#"><u>JobRequirementAttributeRestriction</u></a>	Yes	The restriction of this attribute. May be null, but defaults to <a href="#"><u>JobRequirementAttributeRestriction.must</u></a> .
<b>value</b>	<b>String</b>	Yes	The value of the required attribute. During POST, if value is missing, blank, or null, do not provide a comparator.

JobRequirementAttributeRestriction

Represents a restriction for a job requirement attribute.

Value	Description
<b>must</b>	

JobRequirementFeaturesMode

Value	Description
<b>OR</b>	
<b>AND</b>	

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

NodeAllocationPolicy

Node Allocation enumeration.

Value	Description
<b>FIRSTSET</b>	

Value	Description
<b>MINGLOBAL</b>	
<b>MINLOCAL</b>	
<b>PLUGIN</b>	
<b>NONE</b>	No node allocation policy is specified. Moab defaults to MINRESOURCE when this is the case.
<b>FIRSTAVAILABLE</b>	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, very fast algorithm.
<b>LASTAVAILABLE</b>	This algorithm selects resources so as to minimize the amount of time after the job and before the trailing reservation. This algorithm is a best fit in time algorithm which minimizes the impact of reservation based node-time fragmentation. It is useful in systems where a large number of reservations (job, standing, or administrative) are in place.
<b>MINRESOURCE</b>	This algorithm prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources which still meet the job's resource constraints are selected.
<b>CPULOAD</b>	Nodes are selected which have the maximum amount of available, unused cpu power, i.e. [# of CPU's] - [CPU load]. Good algorithm for timesharing node systems. This algorithm is only applied to jobs starting immediately. For the purpose of future reservations, the <b>MINRESOURCE</b> algorithm is used.
<b>LOCAL</b>	This will call the locally created <b>contrib</b> node allocation algorithm.
<b>CONTIGUOUS</b>	This algorithm will allocate nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	This algorithm will attempt to allocate the most 'balanced' set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node speed. Thus, if possible, nodes with identical speeds will be allocated to the job. If identical speed nodes cannot be found, the algorithm will allocate the set of nodes with the minimum node speed 'span' or range.

Value	Description
<b>PRIORITY</b>	This algorithm allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity.
<b>FASTEST</b>	This algorithm will select nodes in 'fastest node first' order. Nodes will be selected by node speed if specified. If node speed is not specified, nodes will be selected by processor speed. If neither is specified, nodes will be selected in a random order.
<b>PROCESSORLOAD</b>	Alias for CPULOAD.
<b>NODESPEED</b>	Alias for FASTEST.
<b>INREPORTEDORDER</b>	Alias for FIRSTAVAILABLE.
<b>INREVERSEREPORTEDORDER</b>	Alias for LASTAVAILABLE.
<b>CUSTOMPRIORITY</b>	Alias for PRIORITY.
<b>PROCESSORSPEEDBALANCE</b>	Alias for MAXBALANCE.
<b>MINIMUMCONFIGUREDRESOURCES</b>	Alias for MINRESOURCE.

AllocatedNode

Field Name	Type	POST	Description
<b>name</b>	<b>String</b>	No	
<b>taskCount</b>	<b>Integer</b>	No	

JobResource

Represents counts of dedicated and utilized resources.

Field Name	Type	POST	Description
<b>dedicated</b>	<b>Integer</b>	No	The amount of this resource that has been allocated for running workload.
<b>utilized</b>	<b>Integer</b>	No	The amount of this resource that is currently reported as utilized by resource managers.

JobResourceFailPolicyType

Value	Description
<b>CANCEL</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>NOTIFY</b>	
<b>REQUEUE</b>	

ResourceManager

Field Name	Type	POST	Description
<b>isDestination</b>	<b>Boolean</b>	No	
<b>isSource</b>	<b>Boolean</b>	No	
<b>jobName</b>	<b>String</b>	No	
<b>name</b>	<b>String</b>	No	

JobStateInformation

Field Name	Type	POST	Description
<b>state</b>	<a href="#">JobState</a>	No	
<b>stateExpected</b>	<a href="#">JobState</a>	No	
<b>stateLastUpdatedDate</b>	Date	No	
<b>subState</b>	<a href="#">JobSubState</a>	No	

JobState

Value	Description
<b>Idle</b>	
<b>Starting</b>	
<b>Running</b>	
<b>Removed</b>	
<b>Completed</b>	
<b>Hold</b>	
<b>Deferred</b>	
<b>Vacated</b>	
<b>NotQueued</b>	
<b>Unknown</b>	
<b>Staging</b>	
<b>Suspended</b>	
<b>Blocked</b>	

JobSubState

Value	Description
<b>Epilogue</b>	
<b>Migrated</b>	
<b>Preempted</b>	
<b>Prologue</b>	

JobSystemJobType

Value	Description
<b>generic</b>	
<b>osprovision</b>	
<b>osprovision2</b>	
<b>poweroff</b>	
<b>poweron</b>	
<b>reset</b>	
<b>storage</b>	
<b>vmmap</b>	
<b>vmmigrate</b>	
<b>vmtracking</b>	

JobActionType

Value	Description
<b>DESTROY</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

Related topics

- [Job arrays on page 1545](#)

Fields: Jobs

**i** See the associated [Jobs on page 1547](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	jobs	<a href="#">Permissions on page 1593</a>

Type	Value	Additional information
<b>Hooks filename</b>	jobs.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

API version 3

Job

This class represents a job in the Moab Workload Manager. A job is a request for compute resources (CPUs, memory, storage) with which the requester can do work for a given amount of time. In an HPC environment, this might be a batch script to perform a Monte Carlo simulation. In a cloud environment, this would be a virtual machine and its associated storage. Moab will evaluate the request and assign the requested resources to the requester based on policies, current demand, and other factors in the data center. A job will also usually have some process that Moab starts automatically at the assigned start time. In an HPC environment, this can be starting a batch script on the assigned nodes. In a cloud environment, this can be starting provisioning processes to create the virtual machine and storage and install software on it.

Field Name	Type	POS T	PU T	Description
<b>id</b>	<b>String</b>	No	No	The unique identifier of this job. Note: this field is not user-assigned and is generated by the database.
<b>arrayIndex</b>	<b>Long</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>attributes</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of generic attributes associated with this job.
<b>blocks</b>	<a href="#">Set&lt;JobBlock&gt;</a>	No	No	Reasons the job is blocked from running.
<b>bypassCount</b>	<b>Integer</b>	No	No	The number of times the job has been backfilled.

Field Name	Type	POST	PUT	Description
<b>cancelCount</b>	<b>Integer</b>	No	No	The number of times a job has received a cancel request.
<b>commandFile</b>	<b>String</b>	Yes	No	The path to the file that is executed when this job runs. This is the script that will call all the work of this job.
<b>commandLineArguments</b>	<b>String</b>	Yes	No	The command line arguments passed to the commandFile.
<b>completionCode</b>	<b>Integer</b>	No	No	The exit code from this job.
<b>cpuTime</b>	<b>Long</b>	No	No	CPU usage time in seconds as reported by the resource manager.
<b>credentials</b>	<a href="#"><u>JobCredentials</u></a>	Yes	Yes	The credentials (user and group, for example) associated with this job.
<b>customName</b>	<b>String</b>	Yes	Yes	The user-specified name of this job.
<b>dates</b>	<a href="#"><u>JobDates</u></a>	Yes	Yes	Various dates associated with this job.
<b>deferCount</b>	<b>Integer</b>	No	No	The number of times a job has been deferred.
<b>dependencies</b>	<a href="#"><u>Set&lt;JobDependency&gt;</u></a>	Yes	No	Dependencies that must be fulfilled before the job can start.
<b>description</b>	<b>String</b>	No	No	The description of the job. Can be set only in a job template.

Field Name	Type	POST	PUT	Description
<b>duration</b>	<b>Long</b>	Yes	Yes	The length of time in seconds requested for the job. Note that it is possible to set duration to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>durationActive</b>	<b>Long</b>	No	No	The length of time in seconds the job has been active or running.
<b>durationQueued</b>	<b>Long</b>	No	No	The length of time in seconds the job has been eligible to run in the queue.
<b>durationRemaining</b>	<b>Long</b>	No	No	An estimate of the time remaining, in seconds, before the job will complete.
<b>durationSuspended</b>	<b>Long</b>	No	No	The length of time in seconds the job has been suspended.
<b>emailNotifyAddresses</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of addresses to whom email is sent by the execution server.
<b>emailNotifyTypes</b>	<a href="#"><u>Set&lt;JobEmailNotifyType&gt;</u></a>	Yes	No	The list of email notify types attached to the job.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	No	Setting this field to true tells the Moab Workload Manager to set various variables, if populated, in the job's environment.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	No	The list of environment variables for this job.
<b>epilogScript</b>	<b>String</b>	Yes	No	The path to the TORQUE epilog script.

Field Name	Type	POST	PUT	Description
<b>flags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Yes	Yes	The flags that are set on this job.
<b>holdDate</b>	<b>Date</b>	No	No	The date the most recent hold was placed on the job.
<b>holdReason</b>	<a href="#">JobHoldReason</a>	No	No	The reason the job is on hold.
<b>holds</b>	<a href="#">Set&lt;JobHoldType&gt;</a>	Yes	Yes	The holds that are set on the job. The "User" hold type is valid during POST.
<b>initialWorkingDirectory</b>	<b>String</b>	Yes	No	The path to the directory in which the job will be started.
<b>isActive</b>	<b>Boolean</b>	No	No	True if the job is active, false if the job is complete.
<b>jobGroup</b>	<b>String</b>	Yes	No	The job group to which this job belongs (different from credentials.group).
<b>masterNode</b>	<a href="#">DomainProxy</a>	No	No	The first node in the list of allocated nodes for this job. For TORQUE jobs, this represents the "mother superior."
<b>memorySecondsDedicated</b>	<b>Double</b>	No	No	The memory seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>memorySecondsUtilized</b>	<b>Double</b>	No	No	The memory seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.

Field Name	Type	POST	PUT	Description
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	No	Yes	The list of messages associated with the job. The "message" field is valid during PUT.
<b>migrateCount</b>	<b>Integer</b>	No	No	The number of times the job has been migrated.
<b>minimumPreemptTime</b>	<b>Long</b>	No	No	The minimum length of time, in seconds, an active job must be running before it is eligible for preemption.
<b>mwmName</b>	<b>String</b>	No	No	The name of the Moab Workload Manager instance that owns this job.
<b>name</b>	<b>String</b>	No	No	The name of this job. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>nodesExcluded</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The list of nodes that should not be considered for this job.
<b>nodesRequested</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The exact set, superset, or subset of nodes on which this job must run. (See also: <a href="#">nodesRequestedPolicy</a> .)
<b>nodesRequestedPolicy</b>	<a href="#">JobHostListMode</a>	Yes	No	Indicates an exact set, superset, or subset of nodes on which the job must run. Only relevant if nodesRequested is provided. (See also: <a href="#">nodesRequested</a> .)
<b>partitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	No	The list of partitions that this job can access.

Field Name	Type	POST	PUT	Description
<b>partitionAccessListRequested</b>	<b>Set&lt;String&gt;</b>	Yes	Yes	The list of partitions that this job has requested.
<b>preemptCount</b>	<b>Integer</b>	No	No	The number of times the job has been preempted.
<b>priorities</b>	<a href="#">JobPriority</a>	Yes	Yes	The list of priorities for the job.
<b>processorSecondsDedicated</b>	<b>Double</b>	No	No	The processor seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>processorSecondsLimit</b>	<b>Double</b>	No	No	The limit for processorSecondsUtilized.
<b>processorSecondsUtilized</b>	<b>Double</b>	No	No	The processor seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>prologScript</b>	<b>String</b>	Yes	No	The path to the TORQUE prolog script.
<b>queueStatus</b>	<a href="#">JobQueueStatus</a>	No	No	The status of the job in its queue.
<b>rejectPolicies</b>	<a href="#">Set&lt;JobRejectPolicy&gt;</a>	No	No	The list of policies enabled when a job is rejected.
<b>requirements</b>	<a href="#">Set&lt;JobRequirement&gt;</a>	Yes	Yes	The list of items required for this job to run. Only <a href="#">JobRequirement.features</a> is valid during PUT.

Field Name	Type	POST	PULL	Description
<b>reservationRequested</b>	<a href="#">DomainProxy</a>	Yes	Yes	The reservation that the job requested.
<b>resourceFailPolicy</b>	<a href="#">JobResourceFailPolicy Type</a>	Yes	No	The policy that dictates what should happen to the job if it is running and at least one of the resources it is using fails.
<b>resourceManagerExtension</b>	String	Yes	No	If provided during POST, this string will be added to the resource manager extension section of the job submission. Example: "bandwidth=120;queuejob=false" Note that the delimiter between resourceManagerExtension elements is the semicolon.
<b>resourceManagers</b>	<a href="#">Set&lt;ResourceManager&gt;</a>	No	No	The list of resource managers associated with this job.
<b>rmStandardErrorFilePath</b>	String	No	No	The path to the remote file containing the standard error of the job.
<b>rmStandardOutputFilePath</b>	String	No	No	The path to the remote file containing the standard output of the job.
<b>standardErrorFilePath</b>	String	Yes	No	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	String	Yes	No	The path to the file containing the standard output of the job.
<b>startCount</b>	Integer	No	No	The number of times the job has been started.

Field Name	Type	POST	PUT	Description
<b>states</b>	<a href="#">JobStateInformation</a>	No	No	Information about the state of the job.
<b>submitHost</b>	String	Yes	No	The host from which the job was submitted.
<b>systemJobAction</b>	String	No	No	The action the system job will take.
<b>systemJobType</b>	<a href="#">JobSystemJobType</a>	No	No	The type of system job. In the Moab Cloud Suite, this will usually be "vmtracking" or "generic."
<b>targetedJobAction</b>	<a href="#">JobActionType</a>	No	No	The action that this job is performing on another job.
<b>targetedJobName</b>	String	No	No	The name of the job on which this job is performing the targetedJobAction.
<b>templates</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The list of all job templates to be set on this job.
<b>triggers</b>	Set<String>	No	No	The list of triggers associated with this job.
<b>variables</b>	Map<String, Map>	Yes	Yes	The list of variables that this job owns or sets on completion.
<b>virtualContainers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>virtualMachines</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	No	The list of virtual machines that are allocated to this job.

Field Name	Type	POST	PUT	Description
<b>vmUsagePolicy</b>	<a href="#">VMUsagePolicy</a>	Yes	No	The requested Virtual Machine Usage Policy for this job.

JobBlock

Field Name	Type	POST	PUT	Description
<b>category</b>	<a href="#">JobBlockCategory</a>	No	No	
<b>message</b>	String	No	No	
<b>type</b>	<a href="#">JobBlockType</a>	No	No	

JobBlockCategory

Value	Description
<b>depend</b>	
<b>jobBlock</b>	
<b>migrate</b>	

JobBlockType

Value	Description
<b>ActivePolicy</b>	
<b>BadUser</b>	
<b>Dependency</b>	
<b>EState</b>	

Value	Description
<b>FairShare</b>	
<b>Hold</b>	
<b>IdlePolicy</b>	
<b>LocalPolicy</b>	
<b>NoClass</b>	
<b>NoData</b>	
<b>NoResource</b>	
<b>NoTime</b>	
<b>PartitionAccess</b>	
<b>Priority</b>	
<b>RMSubmissionFailure</b>	
<b>StartDate</b>	
<b>State</b>	
<b>SysLimits</b>	

[JobCredentials](#)

Moab Workload Manager supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	POST	PUT	Description
<b>account</b>	<b>String</b>	Yes	Yes	The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes.
<b>group</b>	<b>String</b>	Yes	No	The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.
<b>jobClass</b>	<b>String</b>	Yes	Yes	The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab.
<b>qos</b>	<b>String</b>	No	No	The quality of service assigned to this job.  The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoS's differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.
<b>qosRequested</b>	<b>String</b>	Yes	Yes	The quality of service requested for this job.

Field Name	Type	POST	PUT	Description
<b>user</b>	<b>String</b>	Yes	No	The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

### JobDates

Field Name	Type	POST	PUT	Description
<b>completedDate</b>	<b>Date</b>	No	No	
<b>createdDate</b>	<b>Date</b>	No	No	
<b>deadlineDate</b>	<b>Date</b>	Yes	No	The deadline for completion of the job.
<b>dispatchedDate</b>	<b>Date</b>	No	No	
<b>earliestRequestedStartDate</b>	<b>Date</b>	Yes	Yes	The job will start no sooner than this date.
<b>earliestStartDate</b>	<b>Date</b>	No	No	
<b>eligibleDate</b>	<b>Date</b>	No	No	
<b>lastCanceledDate</b>	<b>Date</b>	No	No	
<b>lastChargedDate</b>	<b>Date</b>	No	No	
<b>lastPreemptedDate</b>	<b>Date</b>	No	No	
<b>lastUpdatedDate</b>	<b>Date</b>	No	No	
<b>startDate</b>	<b>Date</b>	No	No	
<b>submitDate</b>	<b>Date</b>	No	No	

Field Name	Type	POST	PUT	Description
<b>terminationDate</b>	<b>Date</b>	No	No	

JobDependency

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	No	The name of the object on on which the job is dependent.
<b>type</b>	<a href="#"><u>JobDependencyType</u></a>	Yes	No	The type of job dependency. Only set is valid for POST.
<b>value</b>	<b>String</b>	No	No	

JobDependencyType

Represents the type of a job dependency. For now, only the "set" type is supported.

Value	Description
<b>set</b>	

JobEmailNotifyType

Value	Description
<b>JobStart</b>	
<b>JobEnd</b>	
<b>JobFail</b>	
<b>All</b>	

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.

Value	Description
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.

Value	Description
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldReason

Value	Description
<b>Admin</b>	
<b>NoResources</b>	
<b>SystemLimitsExceeded</b>	
<b>BankFailure</b>	

Value	Description
<b>CannotDebitAccount</b>	
<b>InvalidAccount</b>	
<b>RMFailure</b>	
<b>RMReject</b>	
<b>PolicyViolation</b>	
<b>CredAccess</b>	
<b>CredHold</b>	
<b>PreReq</b>	
<b>Data</b>	
<b>Security</b>	
<b>MissingDependency</b>	

JobHoldType

Value	Description
<b>User</b>	
<b>System</b>	
<b>Batch</b>	
<b>Defer</b>	
<b>All</b>	

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	No	The name of the object.

Message

Field Name	Type	POST	PUT	Description
<b>count</b>	<b>Integer</b>	No	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	No	The date this message was created.
<b>expireDate</b>	<b>Date</b>	No	No	The date this message expires.
<b>message</b>	<b>String</b>	No	Yes	The message itself.

JobHostListMode

Value	Description
<b>superset</b>	
<b>subset</b>	
<b>exactset</b>	

JobPriority

Field Name	Type	POST	PUT	Description
<b>run</b>	<b>Long</b>	No	No	
<b>start</b>	<b>Long</b>	No	No	

Field Name	Type	POST	PUT	Description
<b>system</b>	<b>Long</b>	No	No	
<b>user</b>	<b>Long</b>	Yes	Yes	The user-requested priority for the job. By default, the range is between -1024 and 0. To enable priority range from -1024 to +1023, set <code>ENABLEPOSUSERPRIORITY</code> in the <code>moab.cfg</code> file.

JobQueueStatus

Value	Description
<b>active</b>	
<b>blocked</b>	
<b>completed</b>	
<b>eligible</b>	

JobRejectPolicy

Value	Description
<b>CANCEL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>MAIL</b>	
<b>RETRY</b>	

JobRequirement

Field Name	Type	P O S T	P U T	Description
<b>architecture</b>	<b>String</b>	Ye s	N o	The architecture required by the job.
<b>attributes</b>	<a href="#"><u>Map&lt;String, JobRequirementAttribute&gt;</u></a>	Ye s	N o	Required node attributes with version number support.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	Ye s	The list of node features the job is scheduled against.
<b>featuresExcluded</b>	<b>Set&lt;String&gt;</b>	Ye s	N o	Excluded node features. That is, do not select nodes with these features. (See also: <a href="#"><u>featuresExcludedMode</u></a> .)
<b>featuresExcludedMode</b>	<a href="#"><u>JobRequirementFeaturesMode</u></a>	Ye s	N o	Indicates whether excluded features should be AND'ed or OR'd. The default is AND. Only relevant if featuresExcluded is provided. (See also: <a href="#"><u>featuresExcluded</u></a> .)
<b>featuresRequested</b>	<b>Set&lt;String&gt;</b>	Ye s	N o	Requested node features. (See also: <a href="#"><u>featuresRequestedMode</u></a> .)
<b>featuresRequestedMode</b>	<a href="#"><u>JobRequirementFeaturesMode</u></a>	Ye s	N o	Indicates whether requested features should be AND'ed or OR'd. The default is AND. Only relevant if featuresRequested is provided. (See also: <a href="#"><u>featuresRequested</u></a> .)
<b>image</b>	<b>String</b>	Ye s	N o	The image required by the job.
<b>index</b>	<b>Integer</b>	No	N o	The index of the requirement, starting with 0.
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	No	N o	Generic metrics associated with the job as reported by the resource manager.

Field Name	Type	P O S T	P U T	Description
<b>nodeAccessPolicy</b>	<a href="#">NodeAccessPolicy</a>	Yes	No	Specifies how node resources should be accessed. Note: If the job requirements array has more than one element that contains nodeAccessPolicy, only the first occurrence will be used.
<b>nodeAllocationPolicy</b>	<a href="#">NodeAllocationPolicy</a>	Yes	No	Specifies how node resources should be selected and allocated to the job. Note: If the job requirements array has more than one element that contains nodeAllocationPolicy, only the first occurrence will be used.
<b>nodeCount</b>	<b>Integer</b>	Yes	No	The number of nodes required by the job.
<b>nodeSet</b>	<b>String</b>	Yes	No	<p>The requested node set of the job. This must follow the format <code>SETSELECTION:SETTYPE[:SETLIST]</code></p> <ul style="list-style-type: none"> <li>• SETSELECTION - ANYOF, ONEOF, or FIRSTOF</li> <li>• SETTYPE - FEATURE or VARATTR</li> <li>• SETLIST - For FEATURE, a comma-separated list of features. For VARATTR, a key=value pair.</li> </ul> <p>Examples:</p> <ul style="list-style-type: none"> <li>• ONEOF:FEATURE:fastos,hiprio,bigm em</li> <li>• FIRSTOF:VARATTR:datacenter=Provo :datacenter=SaltLake</li> </ul>
<b>nodes</b>	<a href="#">Set&lt;AllocatedNode&gt;</a>	No	No	Nodes that have been allocated to meet this requirement.

Field Name	Type	P O S T	P U T	Description
<b>reservation</b>	<a href="#"><u>DomainProxy</u></a>	No	No	The allocated reservation (assigned after the job has a reservation).
<b>resourcesPerTask</b>	<a href="#"><u>Map&lt;String, JobResource&gt;</u></a>	Yes	No	Contains requirements for disk, memory, processors, swap, and generic resources. For disk, memory, and swap, the unit is MB. For each resource, the "dedicated" field can be set during POST.
<b>taskCount</b>	<b>Integer</b>	Yes	No	The number of tasks (processors) required by this job.
<b>tasksPerNode</b>	<b>Integer</b>	Yes	No	The number of tasks to map to each node.

[JobRequirementAttribute](#)

Field Name	Type	POST	PUT	Description
<b>comparator</b>	<b>String</b>	Yes	No	The comparison operator. Valid values: <ul style="list-style-type: none"> <li>• &gt;= - Greater than or equal to</li> <li>• &gt; - Greater than</li> <li>• &lt;= - Less than</li> <li>• &lt; - Less than</li> <li>• %= - Equals</li> <li>• %! - Not equals</li> <li>• Null - Defaults to %=</li> <li>• = - (Deprecated) Equivalent to %=</li> </ul>
<b>displayValue</b>	<b>String</b>	Yes	No	The display value for the required attribute.
<b>restriction</b>	<a href="#">JobRequirementAttributeRestriction</a>	Yes	No	The restriction of this attribute. May be null, but defaults to <a href="#">JobRequirementAttributeRestriction.must</a> .
<b>value</b>	<b>String</b>	Yes	No	The value of the required attribute. During POST, if value is missing, blank, or null, do not provide a comparator.

JobRequirementAttributeRestriction

Represents a restriction for a job requirement attribute.

Value	Description
<b>must</b>	

JobRequirementFeaturesMode

Value	Description
<b>OR</b>	
<b>AND</b>	

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

NodeAllocationPolicy

Node Allocation enumeration.

Value	Description
<b>FIRSTSET</b>	

Value	Description
<b>MINGLOBAL</b>	
<b>MINLOCAL</b>	
<b>PLUGIN</b>	
<b>NONE</b>	No node allocation policy is specified. Moab defaults to MINRESOURCE when this is the case.
<b>FIRSTAVAILABLE</b>	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, very fast algorithm.
<b>LASTAVAILABLE</b>	This algorithm selects resources so as to minimize the amount of time after the job and before the trailing reservation. This algorithm is a best fit in time algorithm which minimizes the impact of reservation based node-time fragmentation. It is useful in systems where a large number of reservations (job, standing, or administrative) are in place.
<b>MINRESOURCE</b>	This algorithm prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources which still meet the job's resource constraints are selected.
<b>CPULOAD</b>	Nodes are selected which have the maximum amount of available, unused cpu power, i.e. [# of CPU's] - [CPU load]. Good algorithm for timesharing node systems. This algorithm is only applied to jobs starting immediately. For the purpose of future reservations, the <b>MINRESOURCE</b> algorithm is used.
<b>LOCAL</b>	This will call the locally created <b>contrib</b> node allocation algorithm.
<b>CONTIGUOUS</b>	This algorithm will allocate nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	This algorithm will attempt to allocate the most 'balanced' set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node speed. Thus, if possible, nodes with identical speeds will be allocated to the job. If identical speed nodes cannot be found, the algorithm will allocate the set of nodes with the minimum node speed 'span' or range.

Value	Description
<b>PRIORITY</b>	This algorithm allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity.
<b>FASTEST</b>	This algorithm will select nodes in 'fastest node first' order. Nodes will be selected by node speed if specified. If node speed is not specified, nodes will be selected by processor speed. If neither is specified, nodes will be selected in a random order.
<b>PROCESSORLOAD</b>	Alias for CPULOAD.
<b>NODESPEED</b>	Alias for FASTEST.
<b>INREPORTEDORDER</b>	Alias for FIRSTAVAILABLE.
<b>INREVERSEREPORTEDORDER</b>	Alias for LASTAVAILABLE.
<b>CUSTOMPRIORITY</b>	Alias for PRIORITY.
<b>PROCESSORSPEEDBALANCE</b>	Alias for MAXBALANCE.
<b>MINIMUMCONFIGUREDRESOURCES</b>	Alias for MINRESOURCE.

AllocatedNode

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>taskCount</b>	<b>Integer</b>	No	No	

JobResource

Represents counts of dedicated and utilized resources.

Field Name	Type	POST	PUT	Description
<b>dedicated</b>	<b>Integer</b>	No	No	The amount of this resource that has been allocated for running workload.
<b>utilized</b>	<b>Integer</b>	No	No	The amount of this resource that is currently reported as utilized by resource managers.

JobResourceFailPolicyType

Value	Description
<b>CANCEL</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>NOTIFY</b>	
<b>REQUEUE</b>	

ResourceManager

Field Name	Type	POST	PUT	Description
<b>isDestination</b>	<b>Boolean</b>	No	No	
<b>isSource</b>	<b>Boolean</b>	No	No	
<b>jobName</b>	<b>String</b>	No	No	
<b>name</b>	<b>String</b>	No	No	

JobStateInformation

Field Name	Type	POST	PUT	Description
<b>state</b>	<a href="#">JobState</a>	No	No	
<b>stateExpected</b>	<a href="#">JobState</a>	No	No	
<b>stateLastUpdatedDate</b>	Date	No	No	
<b>subState</b>	<a href="#">JobSubState</a>	No	No	

JobState

Value	Description
<b>Idle</b>	
<b>Starting</b>	
<b>Running</b>	
<b>Removed</b>	
<b>Completed</b>	
<b>Hold</b>	
<b>Deferred</b>	
<b>Vacated</b>	
<b>NotQueued</b>	
<b>Unknown</b>	
<b>Staging</b>	
<b>Suspended</b>	
<b>Blocked</b>	

JobSubState

Value	Description
<b>Epilogue</b>	
<b>Migrated</b>	
<b>Preempted</b>	
<b>Prologue</b>	

JobSystemJobType

Value	Description
<b>generic</b>	
<b>osprovision</b>	
<b>osprovision2</b>	
<b>poweroff</b>	
<b>poweron</b>	
<b>reset</b>	
<b>storage</b>	
<b>vmmap</b>	
<b>vmmigrate</b>	
<b>vmtracking</b>	

JobActionType

Value	Description
<b>DESTROY</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

API version 2

Job

This class represents a job in the Moab Workload Manager. A job is a request for compute resources (CPUs, memory, storage) with which the requester can do work for a given amount of time. In an HPC environment, this might be a batch script to perform a Monte Carlo simulation. In a cloud environment, this would be a virtual machine and its associated storage. Moab will evaluate the request and assign the requested resources to the requester based on policies, current demand, and other factors in the data center. A job will also usually have some process that Moab starts automatically at the assigned start time. In an HPC environment, this can be starting a batch script on the assigned nodes. In a cloud environment, this can be starting provisioning processes to create the virtual machine and storage and install software on it.

Field Name	Type	POS T	PU T	Description
<b>id</b>	<b>String</b>	No	No	The unique identifier of this job. Note: this field is not user-assigned and is generated by the database.
<b>arrayIndex</b>	<b>Long</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>attributes</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of generic attributes associated with this job.
<b>blocks</b>	<a href="#">Set&lt;JobBlock&gt;</a>	No	No	Reasons the job is blocked from running.
<b>bypassCount</b>	<b>Integer</b>	No	No	The number of times the job has been backfilled.

Field Name	Type	POST	PUT	Description
<b>cancelCount</b>	<b>Integer</b>	No	No	The number of times a job has received a cancel request.
<b>commandFile</b>	<b>String</b>	Yes	No	The path to the file that is executed when this job runs. This is the script that will call all the work of this job.
<b>commandLineArguments</b>	<b>String</b>	Yes	No	The command line arguments passed to the commandFile.
<b>completionCode</b>	<b>Integer</b>	No	No	The exit code from this job.
<b>cpuTime</b>	<b>Long</b>	No	No	CPU usage time in seconds as reported by the resource manager.
<b>credentials</b>	<a href="#">JobCredentials</a>	Yes	Yes	The credentials (user and group, for example) associated with this job.
<b>customName</b>	<b>String</b>	Yes	Yes	The user-specified name of this job.
<b>dates</b>	<a href="#">JobDates</a>	Yes	Yes	Various dates associated with this job.
<b>deferCount</b>	<b>Integer</b>	No	No	The number of times a job has been deferred.
<b>dependencies</b>	<a href="#">Set&lt;JobDependency&gt;</a>	Yes	No	Dependencies that must be fulfilled before the job can start.
<b>description</b>	<b>String</b>	No	No	The description of the job. Can be set only in a job template.

Field Name	Type	POST	PUR	Description
<b>duration</b>	<b>Long</b>	Yes	Yes	The length of time in seconds requested for the job. Note that it is possible to set duration to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>durationActive</b>	<b>Long</b>	No	No	The length of time in seconds the job has been active or running.
<b>durationQueued</b>	<b>Long</b>	No	No	The length of time in seconds the job has been eligible to run in the queue.
<b>durationRemaining</b>	<b>Long</b>	No	No	An estimate of the time remaining, in seconds, before the job will complete.
<b>durationSuspended</b>	<b>Long</b>	No	No	The length of time in seconds the job has been suspended.
<b>emailNotifyAddresses</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of addresses to whom email is sent by the execution server.
<b>emailNotifyTypes</b>	<a href="#"><u>Set&lt;JobEmailNotifyType&gt;</u></a>	Yes	No	The list of email notify types attached to the job.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	No	Setting this field to true tells the Moab Workload Manager to set various variables, if populated, in the job's environment.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	No	The list of environment variables for this job.
<b>epilogScript</b>	<b>String</b>	Yes	No	The path to the TORQUE epilog script.

Field Name	Type	POST	PUT	Description
<b>flags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Yes	Yes	The flags that are set on this job.
<b>holdDate</b>	<b>Date</b>	No	No	The date the most recent hold was placed on the job.
<b>holdReason</b>	<a href="#">JobHoldReason</a>	No	No	The reason the job is on hold.
<b>holds</b>	<a href="#">Set&lt;JobHoldType&gt;</a>	Yes	Yes	The holds that are set on the job. The "User" hold type is valid during POST.
<b>initialWorkingDirectory</b>	<b>String</b>	Yes	No	The path to the directory in which the job will be started.
<b>isActive</b>	<b>Boolean</b>	No	No	True if the job is active, false if the job is complete.
<b>jobGroup</b>	<b>String</b>	Yes	No	The job group to which this job belongs (different from credentials.group).
<b>masterNode</b>	<a href="#">DomainProxy</a>	No	No	The first node in the list of allocated nodes for this job. For TORQUE jobs, this represents the "mother superior."
<b>memorySecondsDedicated</b>	<b>Double</b>	No	No	The memory seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>memorySecondsUtilized</b>	<b>Double</b>	No	No	The memory seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.

Field Name	Type	POST	PUT	Description
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	No	Yes	The list of messages associated with the job. The "message" field is valid during PUT.
<b>migrateCount</b>	<b>Integer</b>	No	No	The number of times the job has been migrated.
<b>minimumPreemptTime</b>	<b>Long</b>	No	No	The minimum length of time, in seconds, an active job must be running before it is eligible for preemption.
<b>mwmName</b>	<b>String</b>	No	No	The name of the Moab Workload Manager instance that owns this job.
<b>name</b>	<b>String</b>	No	No	The name of this job. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>nodesExcluded</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The list of nodes that should not be considered for this job.
<b>nodesRequested</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The exact set, superset, or subset of nodes on which this job must run. (See also: <a href="#">nodesRequestedPolicy</a> .)
<b>nodesRequestedPolicy</b>	<a href="#">JobHostListMode</a>	Yes	No	Indicates an exact set, superset, or subset of nodes on which the job must run. Only relevant if nodesRequested is provided. (See also: <a href="#">nodesRequested</a> .)
<b>partitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	No	The list of partitions that this job can access.

Field Name	Type	POST	PUT	Description
<b>partitionAccessListRequested</b>	<b>Set&lt;String&gt;</b>	Yes	Yes	The list of partitions that this job has requested.
<b>preemptCount</b>	<b>Integer</b>	No	No	The number of times the job has been preempted.
<b>priorities</b>	<a href="#">JobPriority</a>	Yes	Yes	The list of priorities for the job.
<b>processorSecondsDedicated</b>	<b>Double</b>	No	No	The processor seconds dedicated to the job as reported by its resource manager. Not all resource managers provide this information.
<b>processorSecondsLimit</b>	<b>Double</b>	No	No	The limit for processorSecondsUtilized.
<b>processorSecondsUtilized</b>	<b>Double</b>	No	No	The processor seconds utilized by the job as reported by its resource manager. Not all resource managers provide this information.
<b>prologScript</b>	<b>String</b>	Yes	No	The path to the TORQUE prolog script.
<b>queueStatus</b>	<a href="#">JobQueueStatus</a>	No	No	The status of the job in its queue.
<b>rejectPolicies</b>	<a href="#">Set&lt;JobRejectPolicy&gt;</a>	No	No	The list of policies enabled when a job is rejected.
<b>requirements</b>	<a href="#">Set&lt;JobRequirement&gt;</a>	Yes	Yes	The list of items required for this job to run. Only <a href="#">JobRequirement.features</a> is valid during PUT.

Field Name	Type	POST	PULL	Description
<b>reservationRequested</b>	<a href="#">DomainProxy</a>	Yes	Yes	The reservation that the job requested.
<b>resourceFailPolicy</b>	<a href="#">JobResourceFailPolicy Type</a>	Yes	No	The policy that dictates what should happen to the job if it is running and at least one of the resources it is using fails.
<b>resourceManagerExtension</b>	String	Yes	No	If provided during POST, this string will be added to the resource manager extension section of the job submission. Example: "bandwidth=120;queuejob=false" Note that the delimiter between resourceManagerExtension elements is the semicolon.
<b>resourceManagers</b>	<a href="#">Set&lt;ResourceManager&gt;</a>	No	No	The list of resource managers associated with this job.
<b>rmStandardErrorFilePath</b>	String	No	No	The path to the remote file containing the standard error of the job.
<b>rmStandardOutputFilePath</b>	String	No	No	The path to the remote file containing the standard output of the job.
<b>standardErrorFilePath</b>	String	Yes	No	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	String	Yes	No	The path to the file containing the standard output of the job.
<b>startCount</b>	Integer	No	No	The number of times the job has been started.

Field Name	Type	POST	PUT	Description
<b>states</b>	<a href="#">JobStateInformation</a>	No	No	Information about the state of the job.
<b>submitHost</b>	<b>String</b>	Yes	No	The host from which the job was submitted.
<b>systemJobAction</b>	<b>String</b>	No	No	The action the system job will take.
<b>systemJobType</b>	<a href="#">JobSystemJobType</a>	No	No	The type of system job. In the Moab Cloud Suite, this will usually be "vmtracking" or "generic."
<b>targetedJobAction</b>	<a href="#">JobActionType</a>	No	No	The action that this job is performing on another job.
<b>targetedJobName</b>	<b>String</b>	No	No	The name of the job on which this job is performing the targetedJobAction.
<b>templates</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	The list of all job templates to be set on this job.
<b>triggers</b>	<b>Set&lt;String&gt;</b>	No	No	The list of triggers associated with this job.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The list of variables that this job owns or sets on completion.
<b>virtualContainers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	Yes	No	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>virtualMachines</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	No	The list of virtual machines that are allocated to this job.

Field Name	Type	POST	PUT	Description
<b>vmUsagePolicy</b>	<a href="#">VMUsagePolicy</a>	Yes	No	The requested Virtual Machine Usage Policy for this job.

JobBlock

Field Name	Type	POST	PUT	Description
<b>category</b>	<a href="#">JobBlockCategory</a>	No	No	
<b>message</b>	String	No	No	
<b>type</b>	<a href="#">JobBlockType</a>	No	No	

JobBlockCategory

Value	Description
<b>depend</b>	
<b>jobBlock</b>	
<b>migrate</b>	

JobBlockType

Value	Description
<b>ActivePolicy</b>	
<b>BadUser</b>	
<b>Dependency</b>	
<b>EState</b>	

Value	Description
<b>FairShare</b>	
<b>Hold</b>	
<b>IdlePolicy</b>	
<b>LocalPolicy</b>	
<b>NoClass</b>	
<b>NoData</b>	
<b>NoResource</b>	
<b>NoTime</b>	
<b>PartitionAccess</b>	
<b>Priority</b>	
<b>RMSubmissionFailure</b>	
<b>StartDate</b>	
<b>State</b>	
<b>SysLimits</b>	

[JobCredentials](#)

Moab Workload Manager supports the concept of credentials, which provide a means of attributing policy and resource access to entities such as users and groups. These credentials allow specification of job ownership, tracking of resource usage, enforcement of policies, and many other features.

Field Name	Type	POST	PUT	Description
<b>account</b>	<b>String</b>	Yes	Yes	The account credential is also referred to as the project. This credential is generally associated with a group of users along the lines of a particular project for accounting and billing purposes.
<b>group</b>	<b>String</b>	Yes	No	The group credential represents an aggregation of users. User-to-group mappings are often specified by the operating system or resource manager and typically map to a user's UNIX group ID. However, user-to-group mappings may also be provided by a security and identity management service, or you can specify such directly within Moab.
<b>jobClass</b>	<b>String</b>	Yes	Yes	The concept of the class credential is derived from the resource manager class or queue object. Classes differ from other credentials in that they more directly impact job attributes. In standard HPC usage, a user submits a job to a class and this class imposes a number of factors on the job. The attributes of a class may be specified within the resource manager or directly within Moab.
<b>qos</b>	<b>String</b>	No	No	The quality of service assigned to this job.  The concept of a quality of service (QoS) credential is unique to Moab and is not derived from any underlying concept or peer service. In most cases, the QoS credential is used to allow a site to set up a selection of service levels for end-users to choose from on a long-term or job-by-job basis. QoS's differ from other credentials in that they are centered around special access where this access may allow use of additional services, additional resources, or improved responsiveness. Unique to this credential, organizations may also choose to apply different charge rates to the varying levels of service available within each QoS. As QoS is an internal credential, all QoS configuration occurs within Moab.
<b>qosRequested</b>	<b>String</b>	Yes	Yes	The quality of service requested for this job.

Field Name	Type	POST	PUT	Description
<b>user</b>	<b>String</b>	Yes	No	The user credential is the fundamental credential within a workload manager; each job requires an association with exactly one user. In fact, the user credential is the only required credential in Moab; all others are optional. In most cases, the job's user credential is configured within or managed by the operating system itself, although Moab may be configured to obtain this information from an independent security and identity management service.

JobDates

Field Name	Type	POST	PUT	Description
<b>completedDate</b>	<b>Date</b>	No	No	
<b>createdDate</b>	<b>Date</b>	No	No	
<b>deadlineDate</b>	<b>Date</b>	Yes	No	The deadline for completion of the job.
<b>dispatchedDate</b>	<b>Date</b>	No	No	
<b>earliestRequestedStartDate</b>	<b>Date</b>	Yes	Yes	The job will start no sooner than this date.
<b>earliestStartDate</b>	<b>Date</b>	No	No	
<b>eligibleDate</b>	<b>Date</b>	No	No	
<b>lastCanceledDate</b>	<b>Date</b>	No	No	
<b>lastChargedDate</b>	<b>Date</b>	No	No	
<b>lastPreemptedDate</b>	<b>Date</b>	No	No	
<b>lastUpdatedDate</b>	<b>Date</b>	No	No	
<b>startDate</b>	<b>Date</b>	No	No	
<b>submitDate</b>	<b>Date</b>	No	No	

Field Name	Type	POST	PUT	Description
<b>terminationDate</b>	<b>Date</b>	No	No	

JobDependency

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	No	The name of the object on on which the job is dependent.
<b>type</b>	<a href="#"><u>JobDependencyType</u></a>	Yes	No	The type of job dependency. Only set is valid for POST.
<b>value</b>	<b>String</b>	No	No	

JobDependencyType

Represents the type of a job dependency. For now, only the "set" type is supported.

Value	Description
<b>set</b>	

JobEmailNotifyType

Value	Description
<b>JobStart</b>	
<b>JobEnd</b>	
<b>JobFail</b>	
<b>All</b>	

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.

Value	Description
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.

Value	Description
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldReason

Value	Description
<b>Admin</b>	
<b>NoResources</b>	
<b>SystemLimitsExceeded</b>	
<b>BankFailure</b>	

Value	Description
<b>CannotDebitAccount</b>	
<b>InvalidAccount</b>	
<b>RMFailure</b>	
<b>RMReject</b>	
<b>PolicyViolation</b>	
<b>CredAccess</b>	
<b>CredHold</b>	
<b>PreReq</b>	
<b>Data</b>	
<b>Security</b>	
<b>MissingDependency</b>	

JobHoldType

Value	Description
<b>User</b>	
<b>System</b>	
<b>Batch</b>	
<b>Defer</b>	
<b>All</b>	

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	No	The name of the object.

Message

Field Name	Type	POST	PUT	Description
<b>count</b>	<b>Integer</b>	No	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	No	The date this message was created.
<b>expireDate</b>	<b>Date</b>	No	No	The date this message expires.
<b>message</b>	<b>String</b>	No	Yes	The message itself.

JobHostListMode

Value	Description
<b>superset</b>	
<b>subset</b>	
<b>exactset</b>	

JobPriority

Field Name	Type	POST	PUT	Description
<b>run</b>	<b>Long</b>	No	No	
<b>start</b>	<b>Long</b>	No	No	

Field Name	Type	POST	PUT	Description
<b>system</b>	<b>Long</b>	No	No	
<b>user</b>	<b>Long</b>	Yes	Yes	The user-requested priority for the job. By default, the range is between -1024 and 0. To enable priority range from -1024 to +1023, set <code>ENABLEPOSUSERPRIORITY</code> in the <code>moab.cfg</code> file.

JobQueueStatus

Value	Description
<b>active</b>	
<b>blocked</b>	
<b>completed</b>	
<b>eligible</b>	

JobRejectPolicy

Value	Description
<b>CANCEL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>MAIL</b>	
<b>RETRY</b>	

JobRequirement

Field Name	Type	P O S T	P U T	Description
<b>architecture</b>	<b>String</b>	Ye s	N o	The architecture required by the job.
<b>attributes</b>	<a href="#"><u>Map&lt;String, JobRequirementAttribute&gt;</u></a>	Ye s	N o	Required node attributes with version number support.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	Ye s	The list of node features the job is scheduled against.
<b>featuresExcluded</b>	<b>Set&lt;String&gt;</b>	Ye s	N o	Excluded node features. That is, do not select nodes with these features. (See also: <a href="#"><u>featuresExcludedMode</u></a> .)
<b>featuresExcludedMode</b>	<a href="#"><u>JobRequirementFeaturesMode</u></a>	Ye s	N o	Indicates whether excluded features should be AND'ed or OR'd. The default is AND. Only relevant if featuresExcluded is provided. (See also: <a href="#"><u>featuresExcluded</u></a> .)
<b>featuresRequested</b>	<b>Set&lt;String&gt;</b>	Ye s	N o	Requested node features. (See also: <a href="#"><u>featuresRequestedMode</u></a> .)
<b>featuresRequestedMode</b>	<a href="#"><u>JobRequirementFeaturesMode</u></a>	Ye s	N o	Indicates whether requested features should be AND'ed or OR'd. The default is AND. Only relevant if featuresRequested is provided. (See also: <a href="#"><u>featuresRequested</u></a> .)
<b>image</b>	<b>String</b>	Ye s	N o	The image required by the job.
<b>index</b>	<b>Integer</b>	No	N o	The index of the requirement, starting with 0.
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	No	N o	Generic metrics associated with the job as reported by the resource manager.

Field Name	Type	P O S T	P U T	Description
<b>nodeAccessPolicy</b>	<a href="#">NodeAccessPolicy</a>	Yes	No	Specifies how node resources should be accessed. Note: If the job requirements array has more than one element that contains nodeAccessPolicy, only the first occurrence will be used.
<b>nodeAllocationPolicy</b>	<a href="#">NodeAllocationPolicy</a>	Yes	No	Specifies how node resources should be selected and allocated to the job. Note: If the job requirements array has more than one element that contains nodeAllocationPolicy, only the first occurrence will be used.
<b>nodeCount</b>	<b>Integer</b>	Yes	No	The number of nodes required by the job.
<b>nodeSet</b>	<b>String</b>	Yes	No	<p>The requested node set of the job. This must follow the format SETSELECTION:SETTYPE[:SETLIST]</p> <ul style="list-style-type: none"> <li>• SETSELECTION - ANYOF, ONEOF, or FIRSTOF</li> <li>• SETTYPE - FEATURE or VARATTR</li> <li>• SETLIST - For FEATURE, a comma-separated list of features. For VARATTR, a key=value pair.</li> </ul> <p>Examples:</p> <ul style="list-style-type: none"> <li>• ONEOF:FEATURE:fastos,hiprio,bigm em</li> <li>• FIRSTOF:VARATTR:datacenter=Provo :datacenter=SaltLake</li> </ul>
<b>nodes</b>	<a href="#">Set&lt;AllocatedNode&gt;</a>	No	No	Nodes that have been allocated to meet this requirement.

Field Name	Type	P O S T	P U T	Description
<b>reservation</b>	<a href="#">DomainProxy</a>	No	No	The allocated reservation (assigned after the job has a reservation).
<b>resourcesPerTask</b>	<a href="#">Map&lt;String, JobResource&gt;</a>	Yes	No	Contains requirements for disk, memory, processors, swap, and generic resources. For disk, memory, and swap, the unit is MB. For each resource, the "dedicated" field can be set during POST.
<b>taskCount</b>	<b>Integer</b>	Yes	No	The number of tasks (processors) required by this job.
<b>tasksPerNode</b>	<b>Integer</b>	Yes	No	The number of tasks to map to each node.

[JobRequirementAttribute](#)

Field Name	Type	POST	PUT	Description
<b>comparator</b>	<b>String</b>	Yes	No	The comparison operator. Valid values: <ul style="list-style-type: none"> <li>• &gt;= - Greater than or equal to</li> <li>• &gt; - Greater than</li> <li>• &lt;= - Less than</li> <li>• &lt; - Less than</li> <li>• %= - Equals</li> <li>• %! - Not equals</li> <li>• Null - Defaults to %=</li> <li>• = - (Deprecated) Equivalent to %=</li> </ul>
<b>displayValue</b>	<b>String</b>	Yes	No	The display value for the required attribute.
<b>restriction</b>	<a href="#">JobRequirementAttributeRestriction</a>	Yes	No	The restriction of this attribute. May be null, but defaults to <a href="#">JobRequirementAttributeRestriction.must</a> .
<b>value</b>	<b>String</b>	Yes	No	The value of the required attribute. During POST, if value is missing, blank, or null, do not provide a comparator.

JobRequirementAttributeRestriction

Represents a restriction for a job requirement attribute.

Value	Description
<b>must</b>	

JobRequirementFeaturesMode

Value	Description
<b>OR</b>	
<b>AND</b>	

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

NodeAllocationPolicy

Node Allocation enumeration.

Value	Description
<b>FIRSTSET</b>	

Value	Description
<b>MINGLOBAL</b>	
<b>MINLOCAL</b>	
<b>PLUGIN</b>	
<b>NONE</b>	No node allocation policy is specified. Moab defaults to MINRESOURCE when this is the case.
<b>FIRSTAVAILABLE</b>	Simple first come, first served algorithm where nodes are allocated in the order they are presented by the resource manager. This is a very simple, very fast algorithm.
<b>LASTAVAILABLE</b>	This algorithm selects resources so as to minimize the amount of time after the job and before the trailing reservation. This algorithm is a best fit in time algorithm which minimizes the impact of reservation based node-time fragmentation. It is useful in systems where a large number of reservations (job, standing, or administrative) are in place.
<b>MINRESOURCE</b>	This algorithm prioritizes nodes according to the configured resources on each node. Those nodes with the fewest configured resources which still meet the job's resource constraints are selected.
<b>CPULOAD</b>	Nodes are selected which have the maximum amount of available, unused cpu power, i.e. [# of CPU's] - [CPU load]. Good algorithm for timesharing node systems. This algorithm is only applied to jobs starting immediately. For the purpose of future reservations, the <b>MINRESOURCE</b> algorithm is used.
<b>LOCAL</b>	This will call the locally created <b>contrib</b> node allocation algorithm.
<b>CONTIGUOUS</b>	This algorithm will allocate nodes in contiguous (linear) blocks as required by the Compaq RMS system.
<b>MAXBALANCE</b>	This algorithm will attempt to allocate the most 'balanced' set of nodes possible to a job. In most cases, but not all, the metric for balance of the nodes is node speed. Thus, if possible, nodes with identical speeds will be allocated to the job. If identical speed nodes cannot be found, the algorithm will allocate the set of nodes with the minimum node speed 'span' or range.

Value	Description
<b>PRIORITY</b>	This algorithm allows a site to specify the priority of various static and dynamic aspects of compute nodes and allocate them with preference for higher priority nodes. It is highly flexible allowing node attribute and usage information to be combined with reservation affinity.
<b>FASTEST</b>	This algorithm will select nodes in 'fastest node first' order. Nodes will be selected by node speed if specified. If node speed is not specified, nodes will be selected by processor speed. If neither is specified, nodes will be selected in a random order.
<b>PROCESSORLOAD</b>	Alias for CPULOAD.
<b>NODESPEED</b>	Alias for FASTEST.
<b>INREPORTEDORDER</b>	Alias for FIRSTAVAILABLE.
<b>INREVERSEREPORTEDORDER</b>	Alias for LASTAVAILABLE.
<b>CUSTOMPRIORITY</b>	Alias for PRIORITY.
<b>PROCESSORSPEEDBALANCE</b>	Alias for MAXBALANCE.
<b>MINIMUMCONFIGUREDRESOURCES</b>	Alias for MINRESOURCE.

AllocatedNode

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>taskCount</b>	<b>Integer</b>	No	No	

JobResource

Represents counts of dedicated and utilized resources.

Field Name	Type	POST	PUT	Description
<b>dedicated</b>	<b>Integer</b>	No	No	The amount of this resource that has been allocated for running workload.
<b>utilized</b>	<b>Integer</b>	No	No	The amount of this resource that is currently reported as utilized by resource managers.

JobResourceFailPolicyType

Value	Description
<b>CANCEL</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>IGNORE</b>	
<b>NOTIFY</b>	
<b>REQUEUE</b>	

ResourceManager

Field Name	Type	POST	PUT	Description
<b>isDestination</b>	<b>Boolean</b>	No	No	
<b>isSource</b>	<b>Boolean</b>	No	No	
<b>jobName</b>	<b>String</b>	No	No	
<b>name</b>	<b>String</b>	No	No	

JobStateInformation

Field Name	Type	POST	PUT	Description
<b>state</b>	<a href="#">JobState</a>	No	No	
<b>stateExpected</b>	<a href="#">JobState</a>	No	No	
<b>stateLastUpdatedDate</b>	Date	No	No	
<b>subState</b>	<a href="#">JobSubState</a>	No	No	

JobState

Value	Description
<b>Idle</b>	
<b>Starting</b>	
<b>Running</b>	
<b>Removed</b>	
<b>Completed</b>	
<b>Hold</b>	
<b>Deferred</b>	
<b>Vacated</b>	
<b>NotQueued</b>	
<b>Unknown</b>	
<b>Staging</b>	
<b>Suspended</b>	
<b>Blocked</b>	

JobSubState

Value	Description
<b>Epilogue</b>	
<b>Migrated</b>	
<b>Preempted</b>	
<b>Prologue</b>	

JobSystemJobType

Value	Description
<b>generic</b>	
<b>osprovision</b>	
<b>osprovision2</b>	
<b>poweroff</b>	
<b>poweron</b>	
<b>reset</b>	
<b>storage</b>	
<b>vmmap</b>	
<b>vmmigrate</b>	
<b>vmtracking</b>	

JobActionType

Value	Description
<b>DESTROY</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

## API version 1

JobVersion1

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of the job.
<b>account</b>	<b>String</b>	Yes	Yes	The account under which this job runs for billing purposes.
<b>activeDuration</b>	<b>Long</b>	No	No	The duration in seconds the job has spent active or running.
<b>allocatedNodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The nodes that are allocated to this job.
<b>allocatedVMs</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The list of VMs that are allocated to this job.
<b>arrayIndex</b>	<b>Long</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the index of this job in the array. For example, if this job is <code>Moab.1[2]</code> , the array index would be 2.
<b>arrayMasterName</b>	<b>String</b>	No	No	If this job is a sub-job of a <a href="#">JobArray</a> , this field contains the name of the job array master. For example, if this job is <code>Moab.1[2]</code> , the array master name would be <code>Moab.1</code> .
<b>arraySubJobs</b>	<a href="#">Set&lt;JobArraySubJob&gt;</a>	No	No	If this job is the job array master, this field contains information concerning the array's sub-jobs.
<b>blockReason</b>	<a href="#">JobBlockReasonVersion1</a>	No	No	The reason the job is blocked.
<b>bypass</b>	<b>Integer</b>	No	No	The number of times the job has been backfilled.

Field Name	Type	PO ST	P U T	Description
<b>commandFile</b>	<b>String</b>	Yes	No	The path to the file that is executed when the job runs. This is the script that will call all the work of the job.
<b>commandLineArguments</b>	<b>String</b>	Yes	No	The command line arguments passed in when the job is run.
<b>completionCode</b>	<b>Integer</b>	No	No	The exit code from the job.
<b>completionDate</b>	<b>Date</b>	No	No	The date the job completed. If null, the job hasn't completed yet.
<b>dedicatedProcessorSeconds</b>	<b>Double</b>	No	No	Number of processor seconds dedicated to the job.
<b>dependencies</b>	<a href="#">Set&lt;JobDependencyVersion1&gt;</a>	Yes	No	The list of dependencies for this job. Valid during POST, but only JobDependencyTypeVersion1.SET dependencies are supported. Not defined during GET.
<b>destinationRmJobId</b>	<b>String</b>	No	No	The ID of the job as known by the destination resource manager.
<b>durationRequested</b>	<b>Long</b>	Yes	Yes	The amount of time (in seconds) requested for the job. Note that it is possible to set durationRequested to "INFINITY" if the AllowInfiniteJobs flag is set on the scheduler in the moab.cfg.
<b>earliestStartDate</b>	<b>Date</b>	No	No	Is usually the same as <a href="#">earliestStartDateRequested</a> . Reported by Moab and cannot be changed by the user.

Field Name	Type	POST	PUT	Description
<b>earliestStartDateRequested</b>	<b>Date</b>	Yes	Yes	Used in job creation or modification to set the minimum start time.
<b>effectivePartitionAccessList</b>	<b>Set&lt;String&gt;</b>	No	No	The list of partitions that this job can access.
<b>effectiveQueueDuration</b>	<b>Long</b>	No	No	The duration in seconds the job has been eligible to run in the queue.
<b>emailNotifyTypes</b>	<a href="#">Set&lt;JobEmailNotifyTypeVersion1&gt;</a>	Yes	No	The set of email notify types attached to the job.
<b>emailNotifyUsers</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of users to whom email is sent by the execution server.
<b>environmentRequested</b>	<b>Boolean</b>	Yes	No	Setting this field to true tells Moab to set various variables, if populated, to the job's environment. Not defined during GET.
<b>environmentVariables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	No	The set of environment variables for this job.
<b>expectedState</b>	<a href="#">JobStateVersion1</a>	No	No	The expected state of the job based on scheduler action.
<b>flags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Yes	Yes	The flags that are set on this job.
<b>genericAttributes</b>	<b>Set&lt;String&gt;</b>	No	No	The list of generic attributes the job has requested.
<b>group</b>	<b>String</b>	Yes	No	The group under which the job is run.

Field Name	Type	POST	PUT	Description
<b>holds</b>	<a href="#">Set&lt;JobHoldTypeVersion1&gt;</a>	Yes	Yes	The holds that are set on the job. During POST, only the USER hold type is supported.
<b>hosts</b>	Set<String>	Yes	No	The set of hosts/nodes the job is running on.
<b>initialWorkingDirectory</b>	String	Yes	No	The path to the directory in which the job will be started.
<b>latestCompletedDateRequested</b>	Date	No	No	The latest completion date requested by the job.
<b>masterHost</b>	String	No	No	The node/host that the primary task will run on.
<b>memoryRequested</b>	Integer	No	No	The amount of memory in MB requested by the job.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	No	Yes	The list of system and user messages.
<b>name</b>	String	Yes	Yes	The user-specified name of the job.
<b>os</b>	String	Yes	No	The operating system required for the job.
<b>partitionAccessList</b>	Set<String>	No	No	The list of partitions that this job has requested.
<b>qos</b>	String	No	No	The QoS under which the job is running.
<b>qosRequested</b>	String	Yes	Yes	The QoS the job requested. This may not be the QoS the job is given based on policies.

Field Name	Type	POST	PUT	Description
<b>queue</b>	<b>String</b>	Yes	Yes	The class or queue in which the job is running.
<b>queueStatus</b>	<a href="#">JobQueueStatusVersion1</a>	No	No	The status of the job in its queue.
<b>requirements</b>	<a href="#">Set&lt;JobRequirementVersion1&gt;</a>	Yes	Yes	The list of items required for this job to run. Only <a href="#">JobRequirementVersion1.requiredPartition</a> is valid during PUT.
<b>reservationRequested</b>	<b>String</b>	Yes	Yes	The reservation that the job requested.
<b>reservationStartDate</b>	<b>Date</b>	No	No	The start date of the reservation in which the job is running.
<b>rmExtension</b>	<b>String</b>	No	No	The resource manager extension String that describes information passed to the the resource manager accounting for the job.
<b>rmName</b>	<b>String</b>	No	No	The name of the resource manager that is reporting this job.
<b>rmStandardErrorFilePath</b>	<b>String</b>	No	No	The path to the remote file containing the standard error of the job.
<b>rmStandardInputFilePath</b>	<b>String</b>	No	No	The path to the file containing the standard input of the job.
<b>rmStandardOutputFilePath</b>	<b>String</b>	No	No	The path to the file containing the standard output of the job.
<b>runPriority</b>	<b>Long</b>	No	No	The priority to start running the job.

Field Name	Type	PO ST	P U T	Description
<b>sourceRmJobId</b>	<b>String</b>	No	No	The ID of the job as known by the source resource manager.
<b>standardErrorFilePath</b>	<b>String</b>	Yes	No	The path to the file containing the standard error of the job.
<b>standardOutputFilePath</b>	<b>String</b>	Yes	No	The path to the file containing the standard output of the job.
<b>startCount</b>	<b>Integer</b>	No	No	The number of times the job has been started.
<b>startDate</b>	<b>Date</b>	No	No	The date the job started.
<b>startPriority</b>	<b>Long</b>	No	No	The priority used when the job is first starting.
<b>state</b>	<a href="#">JobStateVersion1</a>	No	No	The job's state.
<b>submitDate</b>	<b>Date</b>	No	No	The time when this job was submitted to Moab.
<b>submitHost</b>	<b>String</b>	Yes	No	The host from which the job was submitted.
<b>suspendDuration</b>	<b>Long</b>	No	No	The duration for which the job has been suspended.
<b>systemPriority</b>	<b>Long</b>	No	No	Administrator-specified system priority level.
<b>templateList</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of all job templates to be set on this job. Not defined during GET.
<b>trigger</b>	<b>String</b>	No	Ye s	Add this trigger to the specified job. Not defined during GET.

Field Name	Type	POST	PUT	Description
<b>user</b>	<b>String</b>	Yes	No	The user that is running the job.
<b>userPriority</b>	<b>Long</b>	Yes	Yes	The user-specified priority for the job.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The set of variables this job "owns" or sets on completion.
<b>virtualContainers</b>	<a href="#"><u>Set&lt;VirtualContainer&gt;</u></a>	Yes	No	When submitting this job, add it to the specified existing virtual container. Valid during POST, but only one virtual container can be specified.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	Yes	No	The requested VM Usage Policy for this job.

[DomainProxyVersion1](#)

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The id of the object.

[JobArraySubJob](#)

Represents information about a job array sub-job. This is used in retrieving Job information only. See [JobArray](#) for the fields used when creating a job array. This class exists only in MWS API version 1.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The ID of this sub-job.
<b>arrayIndex</b>	<b>Long</b>	No	No	The index of this sub-job. For example, if the job ID is <code>Moab.1[4]</code> , the array index would be 4.
<b>state</b>	<a href="#"><u>JobStateVersion1</u></a>	No	No	The current state of this sub-job.

JobStateVersion1

Value	Description
<b>IDLE</b>	Eligible according to all resource manager constraints.
<b>STARTING</b>	Job launching, executing prolog.
<b>RUNNING</b>	Job is executing.
<b>REMOVED</b>	Job canceled before executing.
<b>COMPLETED</b>	Job successfully completed execution.
<b>HOLD</b>	Job is blocked by hold.
<b>DEFERRED</b>	
<b>VACATED</b>	Job canceled after partial execution.
<b>NOT_QUEUED</b>	Job not eligible for execution.
<b>UNKNOWN</b>	
<b>STAGING</b>	Staging of input/output data is currently underway.
<b>SUSPENDED</b>	
<b>BLOCKED</b>	State used only by database for job that is idle but has a block reason.

JobBlockReasonVersion1

Field Name	Type	POST	PUT	Description
<b>message</b>	<b>String</b>	No	No	The message associated with the the reason the job is blocked.
<b>type</b>	<a href="#"><u>JobBlockReasonTypeVersion1</u></a>	No	No	The reason type.

JobBlockReasonTypeVersion1

Value	Description
<b>ACTIVEPOLICY</b>	
<b>BADUSER</b>	
<b>DEPENDENCY</b>	
<b>ESTATE</b>	
<b>FAIRSHARE</b>	
<b>HOLD</b>	
<b>IDLEPOLICY</b>	
<b>LOCALPOLICY</b>	
<b>NOCLASS</b>	
<b>NODATA</b>	
<b>NONE</b>	
<b>NORESOURCE</b>	
<b>NOTIME</b>	
<b>PARTITIONACCESS</b>	
<b>PRIORITY</b>	
<b>RMSUBMISSIONFAILURE</b>	
<b>STARTDATE</b>	
<b>STATE</b>	
<b>SYSLIMITS</b>	

JobDependencyVersion1

Field Name	Type	POST	PUT	Description
<b>dependentJobsCount</b>	<b>Integer</b>	No	No	The number of dependent jobs.
<b>type</b>	<a href="#"><u>JobDependencyTypeVersion1</u></a>	No	No	The dependency type.

JobDependencyTypeVersion1

Value	Description
<b>JOBSTART</b>	Job may start at any time after specified jobs have started execution.
<b>JOBSUCCESSFULCOMPLETE</b>	Job may be start at any time after all specified jobs have successfully completed.
<b>JOBFAILEDCOMPLETE</b>	Job may start at any time after any specified jobs have completed unsuccessfully.
<b>JOBCOMPLETE</b>	Job may start at any time after all specified jobs have completed regardless of completion status.
<b>BEFORE</b>	Job may start at any time before specified jobs have started execution. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREANY</b>	Job may start at any time before all specified jobs have completed regardless of completion status. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREOK</b>	Job may start at any time before all specified jobs have successfully completed. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFORENOTOK</b>	Job may start at any time before any specified jobs have completed unsuccessfully. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>HIBERNATE</b>	Job was set to Hibernate mode.

Value	Description
<b>SYNCWITH</b>	Job will wait until it can start simultaneously with a master job
<b>SYNCCOUNT</b>	This job will wait until it can start simultaneously with synccount jobs of type syncwith that have all specified this synccount job is their master job.
<b>SET</b>	Job will wait until a variable on a Moab object is set before starting.

JobEmailNotifyTypeVersion1

Value	Description
<b>END</b>	An email will be sent out when the job successfully ends.
<b>FAIL</b>	An email will be sent out when the job fails.
<b>START</b>	An email will be sent out when the job starts.

JobStateVersion1

Value	Description
<b>IDLE</b>	Eligible according to all resource manager constraints.
<b>STARTING</b>	Job launching, executing prolog.
<b>RUNNING</b>	Job is executing.
<b>REMOVED</b>	Job canceled before executing.
<b>COMPLETED</b>	Job successfully completed execution.
<b>HOLD</b>	Job is blocked by hold.
<b>DEFERRED</b>	
<b>VACATED</b>	Job canceled after partial execution.

Value	Description
<b>NOT_QUEUED</b>	Job not eligible for execution.
<b>UNKNOWN</b>	
<b>STAGING</b>	Staging of input/output data is currently underway.
<b>SUSPENDED</b>	
<b>BLOCKED</b>	State used only by database for job that is idle but has a block reason.

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.

Value	Description
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.

Value	Description
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.

Value	Description
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobHoldTypeVersion1

Value	Description
<b>NONE</b>	The job does not have a hold.
<b>USER</b>	The user has manually specified a hold on a job.
<b>SYSTEM</b>	Moab has put a hold on the job.
<b>BATCH</b>	The batch queue has placed a hold on the job.
<b>DEFER</b>	The job has been deferred.
<b>ALL</b>	The job has all holds applied to it.

MessageVersion1

Field Name	Type	POST	PUT	Description
<b>author</b>	<b>String</b>	No	No	The author of the message.
<b>creationTime</b>	<b>Date</b>	No	No	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	No	No	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	No	No	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	No	Yes	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	No	No	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	No	No	An optional priority that can be attached to the comment.

JobQueueStatusVersion1

Value	Description
<b>ACTIVE</b>	A job is actively running in a queue.
<b>BLOCKED</b>	A job has been blocked because of policy violation or resource requirements cannot be met.
<b>COMPLETED</b>	A job has completed running.
<b>ELIGIBLE</b>	A job is eligible to run, but has not started yet.
<b>NONE</b>	A job's state type is unknown.

JobRequirementVersion1

Field Name	Type	POST	PUT	Description
<b>allocatedNodes</b>	<a href="#"><u>Set&lt;DomainProxyVersion1&gt;</u></a>	No	No	The set of nodes to which the requirement is allocated.
<b>allocatedPartition</b>	<b>String</b>	No	No	The partition allocated to the requirement.
<b>genericResources</b>	<b>Map&lt;String, Integer&gt;</b>	Yes	No	The set of generic resources for this job. Each generic resource has a name and an associated amount.

Field Name	Type	POST	PUT	Description
<b>nodeAccessPolicy</b>	<a href="#"><u>NodeAccessPolicy</u></a>	Yes	No	The node access policy the requirement asks for.
<b>preferredNodeFeatures</b>	<b>Set&lt;String&gt;</b>	No	No	The list of preferred node features.
<b>requiredArchitecture</b>	<b>String</b>	Yes	No	The required architecture.
<b>requiredClass</b>	<b>String</b>	No	No	The required class/queue.
<b>requiredDiskPerTask</b>	<b>Integer</b>	Yes	No	Required disk per task, in MB.
<b>requiredMemoryPerTask</b>	<b>Integer</b>	Yes	No	Required memory per task, in MB.
<b>requiredNetwork</b>	<b>String</b>	No	No	The required network.
<b>requiredNodeCountMinimum</b>	<b>Integer</b>	No	No	The minimum number of nodes called for by the requirement.
<b>requiredNodeDisk</b>	<b>Integer</b>	No	No	The required disk space in MB across all nodes.

Field Name	Type	POST	PUT	Description
<b>requiredNodeFeatures</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of required node features.
<b>requiredNodeMemory</b>	<b>Integer</b>	No	No	The required memory in MB across all nodes.
<b>requiredNodeProcessors</b>	<b>Integer</b>	No	No	The required processors needed on a per node basis.
<b>requiredNodeSwap</b>	<b>Integer</b>	No	No	The required swap in MB across all nodes.
<b>requiredPartition</b>	<b>String</b>	Yes	Yes	A colon-separated list that specifies the partition (or partitions) in which the job must run.
<b>requiredProcessorCountMinimum</b>	<b>Integer</b>	Yes	No	The minimum task count for this requirement.
<b>requiredProcessorsPerTask</b>	<b>Integer</b>	Yes	No	Required processors per task.
<b>requiredSwapPerTask</b>	<b>Integer</b>	Yes	No	Required swap per task, in MB.

Field Name	Type	POST	PUT	Description
<b>tasksPerNode</b>	<b>Integer</b>	Yes	No	The number of tasks to map to each node.

### NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

### VirtualContainer

A virtual container is a logical grouping of objects with a shared variable space and applied policies. Containers can hold virtual machines, physical machines, jobs, reservations, and/or nodes and req node sets. Containers can also be nested inside other containers.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this virtual container.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	No	No	The set of access control rules associated with this virtual container.
<b>createDate</b>	<b>Date</b>	No	No	The date/time that the virtual container was created.
<b>creator</b>	<b>String</b>	No	No	The creator of the virtual container.
<b>description</b>	<b>String</b>	Yes	Yes	A user-defined string that acts as a label.
<b>flags</b>	<a href="#">Set&lt;VirtualContainerFlag&gt;</a>	No	Yes	The flags on this virtual container.
<b>jobs</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	Yes	The set of jobs in this virtual container.
<b>nodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	Yes	The set of nodes in this virtual container.
<b>owner</b>	<a href="#">EmbeddedCredential</a>	Yes	Yes	The owner of the virtual container.
<b>reservations</b>	<a href="#">Set&lt;Reservation&gt;</a>	No	Yes	The set of reservations in this virtual container.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	No	Yes	Variables associated with the virtual container.
<b>virtualContainers</b>	<a href="#">Set&lt;VirtualContainer&gt;</a>	No	Yes	The set of virtual containers in this virtual container.
<b>virtualMachines</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	Yes	The set of virtual machines in this virtual container.

[AclRule](#)

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	POST	PUT	Description
<b>affinity</b>	<u><a href="#">AclAffinity</a></u>	No	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <u><a href="#">AclAffinity.POSITIVE</a></u> .
<b>comparator</b>	<u><a href="#">ComparisonOperator</a></u>	No	Yes	The type of comparison to make against the ACL object.  Defaults to <u><a href="#">ComparisonOperator.EQUAL</a></u> .
<b>type</b>	<u><a href="#">AclType</a></u>	No	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	No	Yes	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

AcType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported

Value	Description
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

[VirtualContainerFlag](#)

This enumeration specifies the flag types of a virtual container.

Value	Description
<b>DESTROYOBJECTS</b>	Destroy reservations, jobs, and virtual machines in virtual container when the virtual container is destroyed.
<b>DESTROYWHENEMPTY</b>	Destroy virtual container when it contains no objects.
<b>DELETING</b>	Virtual container has started removal process -- might be waiting on workflows, etc. to finish.
<b>HASSTARTED</b>	Virtual container has jobs that have started -- workflows only.
<b>HOLDJOBS</b>	Virtual container will place a hold on jobs that are submitted to it while this flag is set.
<b>WORKFLOW</b>	Virtual container for a workflow -- maximum of one workflow virtual container per workflow.

[EmbeddedCredential](#)

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>type</b>	<a href="#"><u>CredentialType</u></a>	No	No	

[CredentialType](#)

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

[Reservation](#)

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Yes	No	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Yes	No	Accountable Group.

Field Name	Type	POST	PUT	Description
<b>accountingQOS</b>	<b>String</b>	Yes	No	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Yes	No	Accountable User.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	Yes	No	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	No	No	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	No	No	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	No	No	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Yes	No	Reservation's comments or description.
<b>creationDate</b>	<b>Date</b>	No	No	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	Yes	No	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	Yes	No	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.

Field Name	Type	POST	PUT	Description
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	No	No	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	Yes	No	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	No	No	Global reservation ID.
<b>hostListExpression</b>	<b>String</b>	Yes	No	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	Yes	No	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	No	No	State whether or not this reservation is currently active.

Field Name	Type	POST	PUT	Description
<b>isTracked</b>	<b>Boolean</b>	No	No	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	Yes	No	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	No	No	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#"><u>Set&lt;MessageVersion1&gt;</u></a>	No	No	Messages for the reservation.
<b>owner</b>	<a href="#"><u>EmbeddedCredential</u></a>	Yes	No	The owner of the reservation
<b>partitionId</b>	<b>String</b>	Yes	No	The ID of the partition this reservation is for.
<b>profile</b>	<b>String</b>	Yes	No	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#"><u>ReservationRequirement</u></a>	Yes	No	The reservation's requirements.
<b>reservationGroup</b>	<b>String</b>	Yes	No	The reservation group to which the reservation belongs.

Field Name	Type	POST	PUT	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	Yes	No	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	<b>Date</b>	Yes	No	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#"><u>ReservationStatistics</u></a>	No	No	The reservation's statistical information.
<b>subType</b>	<b>String</b>	Yes	No	The reservation sub-type.
<b>taskCount</b>	<b>Integer</b>	No	No	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	Yes	No	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	No	No	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	No	No	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The set of variables for this reservation.

ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOverlap</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.
<b>NOVMMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER==ALL+ allowing all jobs and all users access.

Value	Description
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRED</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.

Value	Description
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.
<b>TRIGHASFIRED</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

### ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	POST	PUT	Description
<b>architecture</b>	<b>String</b>	Yes	No	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	No	No	Required feature mode.
<b>memory</b>	<b>Integer</b>	Yes	No	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	No	No	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	No	No	The list of node IDs required for this reservation.
<b>os</b>	<b>String</b>	Yes	No	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Yes	No	Required task count.

### ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	POST	PUT	Description
<b>caps</b>	<b>Double</b>	No	No	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	No	No	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	No	No	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	No	No	The total idle processor-seconds over the life of the reservation.

### Trigger

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	No	No	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#">TriggerActionType</a>	No	No	
<b>blockTime</b>	<b>Date</b>	No	No	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	No	No	
<b>eventType</b>	<a href="#">TriggerEventType</a>	No	No	
<b>expireTime</b>	<b>Date</b>	No	No	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	<b>Date</b>	No	No	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	No	No	
<b>interval</b>	<b>Boolean</b>	No	No	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false

Field Name	Type	POST	PUT	Description
<b>maxRetry</b>	<b>Integer</b>	No	No	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	No	No	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	No	No	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	No	No	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	No	No	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	No	No	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#">TriggerPeriod</a>	No	No	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	No	No	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	No	No	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	POST	PUT	Description
<b>sets</b>	<b>String</b>	No	No	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	No	No	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	No	No	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	No	No	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.

Value	Description
MAIL	
THRESHOLD	
INTERNAL	
EXEC	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
CANCEL	
CHECKPOINT	
CREATE	
END	
EPOCH	
FAIL	
HOLD	
MIGRATE	
MODIFY	
PREEMPT	
STANDING	
START	
THRESHOLD	

TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.

Value	Description
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	<p>By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).</p> <p>NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.</p>

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.

Value	Description
<b>PREFPM</b>	Prefers a physical machine.
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

Related topics

- [Jobs on page 1547](#)

## Fields: Job Templates

 See the associated [Job templates on page 1569](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	job-templates	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	job-templates.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

JobTemplate

This class represents a job template in the Moab Workload Manager. Job templates are used for two primary purposes: (1) to provide a means of generically matching and categorizing jobs, and (2) to provide a means of setting arbitrary default or forced attributes for certain jobs.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this job template.
<b>account</b>	<b>String</b>	The account under which this job will run for billing purposes.
<b>args</b>	<b>String</b>	Command-line arguments that get passed to commandFile.
<b>commandFile</b>	<b>String</b>	The path to the file that is executed when the job runs. This is the script that will actually call all the work of the job. Can be null.
<b>description</b>	<b>String</b>	The description of the job.
<b>durationRequested</b>	<b>Long</b>	The amount of time (in seconds) requested for the job.
<b>genericSystemJob</b>	<b>Boolean</b>	True if this template will instantiate a generic system job.
<b>inheritResources</b>	<b>Boolean</b>	True if jobs instantiated from this template inherit resources.
<b>jobDependencies</b>	<a href="#">Set&lt;JobTemplateDependency&gt;</a>	The list of dependencies for this job template.
<b>jobFlags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Job flags for this template.
<b>jobTemplateFlags</b>	<a href="#">Set&lt;JobTemplateFlag&gt;</a>	Job template flags for this template.
<b>jobTemplateRequirements</b>	<a href="#">Set&lt;JobTemplateRequirement&gt;</a>	The requirements for this job template.

Field Name	Type	Description
<b>priority</b>	<b>Long</b>	Relative job priority.
<b>qos</b>	<b>String</b>	The Quality of Service for the job.
<b>queue</b>	<b>String</b>	The class or queue in which the job will run.
<b>select</b>	<b>Boolean</b>	True if job template can be directly requested by job at submission.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	The trigger that is typically assigned to generic system jobs.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	The virtual machine usage policy.

[JobTemplateDependency](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the template on which this template depends.
<b>type</b>	<a href="#"><u>JobDependencyTypeVersion1</u></a>	The type of the dependency.

[JobDependencyTypeVersion1](#)

Value	Description
<b>JOBSTART</b>	Job may start at any time after specified jobs have started execution.
<b>JOBSUCCESSFULCOMPLETE</b>	Job may be start at any time after all specified jobs have successfully completed.
<b>JOBFAILEDCOMPLETE</b>	Job may start at any time after any specified jobs have completed unsuccessfully.

Value	Description
<b>JOBCOMPLETE</b>	Job may start at any time after all specified jobs have completed regardless of completion status.
<b>BEFORE</b>	Job may start at any time before specified jobs have started execution. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREANY</b>	Job may start at any time before all specified jobs have completed regardless of completion status. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREOK</b>	Job may start at any time before all specified jobs have successfully completed. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFORENOTOK</b>	Job may start at any time before any specified jobs have completed unsuccessfully. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>HIBERNATE</b>	Job was set to Hibernate mode.
<b>SYNCWITH</b>	Job will wait until it can start simultaneously with a master job
<b>SYNCCOUNT</b>	This job will wait until it can start simultaneously with synccount jobs of type syncwith that have all specified this synccount job is their master job.
<b>SET</b>	Job will wait until a variable on a Moab object is set before starting.

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.

Value	Description
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.

Value	Description
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).

Value	Description
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobTemplateFlag

This enumeration specifies the flag types of a job template.

Value	Description
<b>GLOBALRSVACCESS</b>	
<b>HIDDEN</b>	
<b>HWJOB</b>	
<b>PRIVATE</b>	
<b>SYNCJOBID</b>	
<b>TEMPLATEISDYNAMIC</b>	True if the template is dynamic (not specified via moab.cfg).
<b>SELECT</b>	True if a job can select this template.

JobTemplateRequirement

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	The architecture requirement.
<b>diskRequirement</b>	<b>Integer</b>	The amount of disk space required (in MB).
<b>genericResources</b>	<b>Map&lt;String, Integer&gt;</b>	Consumable generic attributes associated with individual nodes or the special pseudo-node global, which provides shared cluster (floating) consumable resources.
<b>nodeAccessPolicy</b>	<a href="#"><u>NodeAccessPolicy</u></a>	The node access policy. Specifies how node resources will be shared by a job.
<b>operatingSystem</b>	<b>String</b>	The operating system requirement.
<b>requiredDiskPerTask</b>	<b>Integer</b>	Disk space (in MB).
<b>requiredFeatures</b>	<b>Set&lt;String&gt;</b>	The features required by this template.
<b>requiredMemoryPerTask</b>	<b>Integer</b>	Memory (in MB).
<b>requiredProcessorsPerTask</b>	<b>Integer</b>	Number of processors.
<b>requiredSwapPerTask</b>	<b>Integer</b>	Swap space (in MB).
<b>taskCount</b>	<b>Integer</b>	The number of tasks required.

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.

Value	Description
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

### TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.

Value	Description
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

## API version 2

JobTemplate

This class represents a job template in the Moab Workload Manager. Job templates are used for two primary purposes: (1) to provide a means of generically matching and categorizing jobs, and (2) to provide a means of setting arbitrary default or forced attributes for certain jobs.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this job template.
<b>account</b>	<b>String</b>	The account under which this job will run for billing purposes.
<b>args</b>	<b>String</b>	Command-line arguments that get passed to commandFile.
<b>commandFile</b>	<b>String</b>	The path to the file that is executed when the job runs. This is the script that will actually call all the work of the job. Can be null.
<b>description</b>	<b>String</b>	The description of the job.
<b>durationRequested</b>	<b>Long</b>	The amount of time (in seconds) requested for the job.
<b>genericSystemJob</b>	<b>Boolean</b>	True if this template will instantiate a generic system job.
<b>inheritResources</b>	<b>Boolean</b>	True if jobs instantiated from this template inherit resources.
<b>jobDependencies</b>	<a href="#">Set&lt;JobTemplateDependency&gt;</a>	The list of dependencies for this job template.
<b>jobFlags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Job flags for this template.
<b>jobTemplateFlags</b>	<a href="#">Set&lt;JobTemplateFlag&gt;</a>	Job template flags for this template.
<b>jobTemplateRequirements</b>	<a href="#">Set&lt;JobTemplateRequirement&gt;</a>	The requirements for this job template.

Field Name	Type	Description
<b>priority</b>	<b>Long</b>	Relative job priority.
<b>qos</b>	<b>String</b>	The Quality of Service for the job.
<b>queue</b>	<b>String</b>	The class or queue in which the job will run.
<b>select</b>	<b>Boolean</b>	True if job template can be directly requested by job at submission.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	The trigger that is typically assigned to generic system jobs.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	The virtual machine usage policy.

[JobTemplateDependency](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the template on which this template depends.
<b>type</b>	<a href="#"><u>JobDependencyTypeVersion1</u></a>	The type of the dependency.

[JobDependencyTypeVersion1](#)

Value	Description
<b>JOBSTART</b>	Job may start at any time after specified jobs have started execution.
<b>JOBSUCCESSFULCOMPLETE</b>	Job may be start at any time after all specified jobs have successfully completed.
<b>JOBFAILEDCOMPLETE</b>	Job may start at any time after any specified jobs have completed unsuccessfully.

Value	Description
<b>JOBCOMPLETE</b>	Job may start at any time after all specified jobs have completed regardless of completion status.
<b>BEFORE</b>	Job may start at any time before specified jobs have started execution. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREANY</b>	Job may start at any time before all specified jobs have completed regardless of completion status. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREOK</b>	Job may start at any time before all specified jobs have successfully completed. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFORENOTOK</b>	Job may start at any time before any specified jobs have completed unsuccessfully. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>HIBERNATE</b>	Job was set to Hibernate mode.
<b>SYNCWITH</b>	Job will wait until it can start simultaneously with a master job
<b>SYNCCOUNT</b>	This job will wait until it can start simultaneously with synccount jobs of type syncwith that have all specified this synccount job is their master job.
<b>SET</b>	Job will wait until a variable on a Moab object is set before starting.

JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.

Value	Description
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.

Value	Description
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).

Value	Description
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

JobTemplateFlag

This enumeration specifies the flag types of a job template.

Value	Description
<b>GLOBALRSVACCESS</b>	
<b>HIDDEN</b>	
<b>HWJOB</b>	
<b>PRIVATE</b>	
<b>SYNCJOBID</b>	
<b>TEMPLATEISDYNAMIC</b>	True if the template is dynamic (not specified via moab.cfg).
<b>SELECT</b>	True if a job can select this template.

JobTemplateRequirement

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	The architecture requirement.
<b>diskRequirement</b>	<b>Integer</b>	The amount of disk space required (in MB).
<b>genericResources</b>	<b>Map&lt;String, Integer&gt;</b>	Consumable generic attributes associated with individual nodes or the special pseudo-node global, which provides shared cluster (floating) consumable resources.
<b>nodeAccessPolicy</b>	<a href="#"><u>NodeAccessPolicy</u></a>	The node access policy. Specifies how node resources will be shared by a job.
<b>operatingSystem</b>	<b>String</b>	The operating system requirement.
<b>requiredDiskPerTask</b>	<b>Integer</b>	Disk space (in MB).
<b>requiredFeatures</b>	<b>Set&lt;String&gt;</b>	The features required by this template.
<b>requiredMemoryPerTask</b>	<b>Integer</b>	Memory (in MB).
<b>requiredProcessorsPerTask</b>	<b>Integer</b>	Number of processors.
<b>requiredSwapPerTask</b>	<b>Integer</b>	Swap space (in MB).
<b>taskCount</b>	<b>Integer</b>	The number of tasks required.

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.

Value	Description
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

### TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.

Value	Description
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

## API version 1

JobTemplate

This class represents a job template in the Moab Workload Manager. Job templates are used for two primary purposes: (1) to provide a means of generically matching and categorizing jobs, and (2) to provide a means of setting arbitrary default or forced attributes for certain jobs.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique identifier for this job template.
<b>account</b>	<b>String</b>	The account under which this job will run for billing purposes.
<b>args</b>	<b>String</b>	Command-line arguments that get passed to commandFile.
<b>commandFile</b>	<b>String</b>	The path to the file that is executed when the job runs. This is the script that will actually call all the work of the job. Can be null.
<b>description</b>	<b>String</b>	The description of the job.
<b>durationRequested</b>	<b>Long</b>	The amount of time (in seconds) requested for the job.
<b>genericSystemJob</b>	<b>Boolean</b>	True if this template will instantiate a generic system job.
<b>inheritResources</b>	<b>Boolean</b>	True if jobs instantiated from this template inherit resources.
<b>jobDependencies</b>	<a href="#">Set&lt;JobTemplateDependency&gt;</a>	The list of dependencies for this job template.
<b>jobFlags</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Job flags for this template.
<b>jobTemplateFlags</b>	<a href="#">Set&lt;JobTemplateFlag&gt;</a>	Job template flags for this template.
<b>jobTemplateRequirements</b>	<a href="#">Set&lt;JobTemplateRequirement&gt;</a>	The requirements for this job template.

Field Name	Type	Description
<b>priority</b>	<b>Long</b>	Relative job priority.
<b>qos</b>	<b>String</b>	The Quality of Service for the job.
<b>queue</b>	<b>String</b>	The class or queue in which the job will run.
<b>select</b>	<b>Boolean</b>	True if job template can be directly requested by job at submission.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	The trigger that is typically assigned to generic system jobs.
<b>vmUsagePolicy</b>	<a href="#"><u>VMUsagePolicy</u></a>	The virtual machine usage policy.

#### [JobTemplateDependency](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	The name of the template on which this template depends.
<b>type</b>	<a href="#"><u>JobDependencyTypeVersion1</u></a>	The type of the dependency.

#### [JobDependencyTypeVersion1](#)

Value	Description
<b>JOBSTART</b>	Job may start at any time after specified jobs have started execution.
<b>JOBSUCCESSFULCOMPLETE</b>	Job may be start at any time after all specified jobs have successfully completed.
<b>JOBFAILEDCOMPLETE</b>	Job may start at any time after any specified jobs have completed unsuccessfully.

Value	Description
<b>JOBCOMPLETE</b>	Job may start at any time after all specified jobs have completed regardless of completion status.
<b>BEFORE</b>	Job may start at any time before specified jobs have started execution. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREANY</b>	Job may start at any time before all specified jobs have completed regardless of completion status. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFOREOK</b>	Job may start at any time before all specified jobs have successfully completed. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>BEFORENOTOK</b>	Job may start at any time before any specified jobs have completed unsuccessfully. NOTE: Only reported to Moab and then reported back. Moab currently cannot internally handle this type of dependency.
<b>HIBERNATE</b>	Job was set to Hibernate mode.
<b>SYNCWITH</b>	Job will wait until it can start simultaneously with a master job
<b>SYNCCOUNT</b>	This job will wait until it can start simultaneously with synccount jobs of type syncwith that have all specified this synccount job is their master job.
<b>SET</b>	Job will wait until a variable on a Moab object is set before starting.

### JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.

Value	Description
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.

Value	Description
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).

Value	Description
<b>DESTROYTEMPLATESUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

#### JobTemplateFlag

This enumeration specifies the flag types of a job template.

Value	Description
<b>GLOBALRSVACCESS</b>	
<b>HIDDEN</b>	
<b>HWJOB</b>	
<b>PRIVATE</b>	
<b>SYNCJOBID</b>	
<b>TEMPLATEISDYNAMIC</b>	True if the template is dynamic (not specified via moab.cfg).
<b>SELECT</b>	True if a job can select this template.

JobTemplateRequirement

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	The architecture requirement.
<b>diskRequirement</b>	<b>Integer</b>	The amount of disk space required (in MB).
<b>genericResources</b>	<b>Map&lt;String, Integer&gt;</b>	Consumable generic attributes associated with individual nodes or the special pseudo-node global, which provides shared cluster (floating) consumable resources.
<b>nodeAccessPolicy</b>	<a href="#"><u>NodeAccessPolicy</u></a>	The node access policy. Specifies how node resources will be shared by a job.
<b>operatingSystem</b>	<b>String</b>	The operating system requirement.
<b>requiredDiskPerTask</b>	<b>Integer</b>	Disk space (in MB).
<b>requiredFeatures</b>	<b>Set&lt;String&gt;</b>	The features required by this template.
<b>requiredMemoryPerTask</b>	<b>Integer</b>	Memory (in MB).
<b>requiredProcessorsPerTask</b>	<b>Integer</b>	Number of processors.
<b>requiredSwapPerTask</b>	<b>Integer</b>	Swap space (in MB).
<b>taskCount</b>	<b>Integer</b>	The number of tasks required.

NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.

Value	Description
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

VMUsagePolicy

This enumeration describes the virtual machine requirements of a job

Value	Description
<b>REQUIREPM</b>	Requires a physical machine.
<b>PREFPM</b>	Prefers a physical machine.

Value	Description
<b>CREATEVM</b>	Creates a virtual machine.
<b>CREATEPERSISTENTVM</b>	Creates a virtual machine that doesn't go away after the job is done.
<b>REQUIREVM</b>	Requires a virtual machine.
<b>PREFVM</b>	Prefers a virtual machine.

Related topics

- [Job templates on page 1569](#)

## Fields: Metric Types

 See the associated [Metric types on page 1571](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	metric-types	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	metric-types.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

API version 3

MetricType

Represents a metric visible and known to Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this metric type.

## API version 2

MetricType

Represents a metric visible and known to Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this metric type.

## API version 1

MetricType

Represents a metric visible and known to Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this metric type.

## Related topics

- [Metric types on page 1571](#)

## Fields: Nodes

 See the associated [Nodes on page 1573](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	nodes	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	nodes.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

Node

This class represents a node in the Moab Workload Manager. Moab recognizes a node as a collection of resources with a particular set of associated attributes. This definition is similar to the traditional notion of a node found in a Linux cluster or supercomputer wherein a node is defined as one or more CPUs, associated memory, and possibly other compute resources such as local disk, swap, network adapters, and software licenses. Additionally, this node is described by various attributes such as an architecture type or operating system. Nodes range in size from small uniprocessor PCs to large symmetric multiprocessing (SMP) systems where a single node may consist of hundreds of CPUs and massive amounts of memory.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier of this node. Note: this field is not user-assigned and is generated by the database.
<b>architecture</b>	<b>String</b>	No	This node's processor architecture.
<b>attributes</b>	<b>Map&lt;String, Map&gt;</b>	No	Attributes is a map of attribute names to tuples (maps) that describe the scheduling attributes of a node. Each tuple should contain the following entries: <ul style="list-style-type: none"> <li>• <b>value</b> - the attribute value</li> <li>• <b>displayValue</b> - the attribute display value</li> </ul>
<b>classes</b>	<b>Set&lt;String&gt;</b>	No	The classes that this node can be scheduled for.
<b>featuresCustom</b>	<b>Set&lt;String&gt;</b>	Yes	The features this node advertises which are customizable at run-time. This can be used to define node sets. (See also: <a href="#">featuresReported</a> .)

Field Name	Type	P UT	Description
<b>featuresReported</b>	<b>Set&lt;String&gt;</b>	No	The features this node advertises which are reported by resource managers or are present in the Moab Workload Manager configuration. This can be used to define node sets. (See also: <a href="#">featuresCustom</a> .)
<b>index</b>	<b>Integer</b>	No	The index for this node as reported by the resource manager.
<b>ipAddress</b>	<b>String</b>	No	This node's IPv4 address.
<b>isHypervisor</b>	<b>Boolean</b>	No	True if the node is a hypervisor, false otherwise. This is based on the <a href="#">NodeOperatingSystemInformation.hypervisorType</a> field. If hypervisorType is present, the node is a hypervisor. If it is null, then it is not a hypervisor.
<b>jobs</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Jobs associated with this node.
<b>lastUpdatedDate</b>	<b>Date</b>	No	The timestamp of the last moment when this node was updated. There is no guarantee that all user modifications to a node would be picked up. This will also be changed every RMPOLLINTERVAL even if a resource manager does not report information on this node.
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	Yes	The list of messages attached to this node. They can be attached by admins, the resource manager layer, or triggers.

Field Name	Type	PUT	Description
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	Yes	<p>Metrics are the measurable, quantitative, and changing aspects of this node. They are used to define workload placement, attach triggers, etc. There are some built-in metrics:</p> <ul style="list-style-type: none"> <li>• <b>speed</b> - A number from 0.0 to 1.0 describing the relative speed of the system for computational tasks. This is a composite metric, and is defined on a per-site basis.</li> <li>• <b>cpuLoad</b> - This is the CPU load on this node. This value is defined at the resource manager layer, but is generally defined on a per-operating system basis. For example, Unix-based OS's use some aspect of the Unix load average, as reported by the resource manager layer, while Windows-based OS's use CPU utilization.</li> </ul>
<b>migrationDisabled</b>	<b>Boolean</b>	No	True if VM migration is disabled on this node.
<b>name</b>	<b>String</b>	No	The name of this node. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>operatingSystem</b>	<a href="#">NodeOperatingSystemInformation</a>	Yes	Describes the current or expected operating system image information for this node. The <code>operatingSystem.image</code> field can be changed using PUT.
<b>partition</b>	<b>String</b>	Yes	The partition this node belongs to.
<b>processorSpeed</b>	<b>Integer</b>	No	The speed, in MHz, or the processors on this node.

Field Name	Type	P U T	Description
<b>profilingEnabled</b>	<b>Boolean</b>	No	Indicates whether historical data gathering and reporting is enabled for this node. This is also controlled by the same setting on the default node (i.e. all nodes). If set to false (default), node statistics are not gathered.
<b>rack</b>	<b>Integer</b>	No	The rack where this node is located in the datacenter/cluster.
<b>reservations</b>	<a href="#"><u>Set&lt;DomainProxy&gt;</u></a>	No	Reservations associated with this node.
<b>resourceManagerMessages</b>	<b>Map&lt;String, Map&gt;</b>	No	The resource manager messages for this node. Each key is the name of a resource manager, and the value is the message that the resource manager has posted onto the node.
<b>resourceManagers</b>	<a href="#"><u>Set&lt;NodeResourceManager&gt;</u></a>	No	The resource managers that are reporting or have previously reported this node. Each object also contains information on the resource manager reports.
<b>resources</b>	<a href="#"><u>Map&lt;String, Resource&gt;</u></a>	No	Contains references of a string representing a resource name to a resource object detailing the amount of the resource that is available, configured, etc. Each key is the name of the resource, which equates to the generic resource identifier or one of "processors", "memory", "disk", or "swap". This name may be used as an id in the resource types web service.
<b>slot</b>	<b>Integer</b>	No	The slot in the rack where this node is located.

Field Name	Type	PUT	Description
<b>states</b>	<a href="#">NodeStateInformation</a>	Yes	This node's state. The states.powerState and states.state fields can be changed using PUT.
<b>triggers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Triggers associated with this node.
<b>type</b>	<a href="#">NodeType</a>	No	The type of this node is governed by the types of resources it offers.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Variables is a map of key-value pairs, synonymous, but not directly related to, environment variables. They provide the mechanism to store arbitrary metadata which is useful to external systems in memory on this node.
<b>virtualContainers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	The set of virtual containers that directly (not recursively) contain this node.
<b>virtualMachines</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Virtual machines associated with this node.

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	PUT	Description
<b>name</b>	<b>String</b>	No	The name of the object.

Message

Field Name	Type	PUT	Description
<b>count</b>	<b>Integer</b>	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	The date this message was created.

Field Name	Type	PUT	Description
<b>expireDate</b>	<b>Date</b>	No	The date this message expires.
<b>message</b>	<b>String</b>	Yes	The message itself.

### NodeOperatingSystemInformation

Describes the current or expected operating system image information for a node.

Field Name	Type	PUT	Description
<b>hypervisorType</b>	<b>String</b>	No	The hypervisor technology that this node uses. May be null if the node is not a hypervisor.
<b>image</b>	<b>String</b>	Yes	The name of the operating system currently running on this node. In cloud mode, this corresponds to the ID or name of an image in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>imageExpected</b>	<b>String</b>	No	The name of the image that was requested to run on this node (i.e. with <code>mnodectl -m os=myOs</code> ). In cloud mode, this corresponds to the ID or name of an image in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>imageLastUpdatedDate</b>	<b>Date</b>	No	The last time the image of this node was modified.
<b>imagesAvailable</b>	<b>Set&lt;String&gt;</b>	No	The list of image names that can be applied to this node. In cloud mode, this corresponds to IDs or names of images in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>virtualMachineImages</b>	<b>Set&lt;String&gt;</b>	No	The list of virtual machine image names the node is capable of supporting. In cloud mode, this corresponds to IDs or names of images in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)

### NodeResourceManager

Field Name	Type	PUT	Description
<b>isMaster</b>	<b>Boolean</b>	No	Indicates whether this resource manager is the "master" of this Node. If true, it means that this resource manager has the final say on all properties reported about this Node. Note that the first resource manager to report a node is the master resource manager.
<b>name</b>	<b>String</b>	No	The name of the resource manager, according to Moab. This name appears in both the RMCFG parameter, and when diagnosing resource managers (e.g. <code>mdiag -R</code> ).
<b>stateReported</b>	<a href="#">NodeState</a>	No	The state reported by this resource manager. See the State section for more details.

### [NodeState](#)

This enumeration tracks the state of a node.

Value	Description
<b>NONE</b>	The node is set to none by the resource manager.
<b>DOWN</b>	The node is not available for workload.
<b>IDLE</b>	The node is available for workload but is not running anything.
<b>BUSY</b>	The node is running workload and cannot accept more.
<b>RUNNING</b>	The node is running workload and can accept more.
<b>DRAINED</b>	The node has been sent the drain request and has no workload on it.
<b>DRAINING</b>	The node has been sent the drain request, but still has workload on it.
<b>FLUSH</b>	The node is being reprovisioned.
<b>RESERVED</b>	The node is being reserved. This is an internal Moab state.
<b>UNKNOWN</b>	The state of the node is unknown.

Resource

Represents counts of resources available, configured, etc.

Field Name	Type	PUT	Description
<b>available</b>	<b>Integer</b>	No	The amount of this resource that is currently available for allocation to workload.
<b>configured</b>	<b>Integer</b>	No	The amount of this resource that is considered possible to schedule. Overcommit specifically applies to this, in other words, <code>configured = overcommitFactor * real</code> .
<b>dedicated</b>	<b>Integer</b>	No	The amount of this resource that has been allocated for running workload. When used in a job submission, this number is the amount of the resource required by the job.
<b>real</b>	<b>Integer</b>	No	The amount of this resource that physically exists on the node. Overcommit specifically doesn't apply to this. Note that overcommit currently only applies to "processors" and "memory", and so, for most cases, real and configured will always be the same.
<b>utilized</b>	<b>Integer</b>	No	The amount of this resource that is currently reported as utilized by resource managers.

NodeStateInformation

Field Name	Type	PUT	Description
<b>powerState</b>	<a href="#">NodePower</a>	Yes	The state of the node's power system, as reported by the RM layer. Modifying the <code>powerState</code> is possible, and, if Moab is configured properly, a request will be made to modify the power state accordingly.
<b>powerStateExpected</b>	<a href="#">NodePower</a>	No	The expected state of the node's power system. If a user has requested that a node be powered off (e.g. by modifying the <code>powerState</code> attribute to <a href="#">NodePower.OFF</a> ), the requested state will be shown in this field until the state change is completed. If there is no pending power change request, this will be null.

Field Name	Type	PUT	Description
<b>state</b>	<a href="#">NodeState</a>	Yes	The scheduling state of the Node, as reported by the resource management layer.
<b>stateExpected</b>	<a href="#">NodeState</a>	No	The scheduling state of the Node, as expected by Moab. For example, Moab may think that a Node is "Busy" because it has allocated all configured resources, but a resource manager may report the state as "Running" based on actual utilization of the resources.
<b>stateLastUpdatedDate</b>	<b>Date</b>	No	A timestamp recording when the state of the Node was last modified.
<b>subState</b>	<b>String</b>	No	A text description of the state of the Node, with the intention of giving more details. Resource Managers may use this field to further describe the state being reported. Resource Managers should provide documented meaning to the possible sub-states that they can report.
<b>subStateLast</b>	<b>String</b>	No	The previous sub-state of the Node as reported by the resource management layer.
<b>subStateLastUpdatedDate</b>	<b>Date</b>	No	A timestamp recording when the sub-state was last modified.

### [NodePower](#)

Represents the various options for a Node's power state.

Value	Description
<b>NONE</b>	
<b>ON</b>	
<b>OFF</b>	

### [NodeType](#)

Represents the type of node as reported by a resource manager.

Value	Description
<b>Compute</b>	
<b>License</b>	
<b>Network</b>	
<b>Storage</b>	

## API version 2

Node

This class represents a node in the Moab Workload Manager. Moab recognizes a node as a collection of resources with a particular set of associated attributes. This definition is similar to the traditional notion of a node found in a Linux cluster or supercomputer wherein a node is defined as one or more CPUs, associated memory, and possibly other compute resources such as local disk, swap, network adapters, and software licenses. Additionally, this node is described by various attributes such as an architecture type or operating system. Nodes range in size from small uniprocessor PCs to large symmetric multiprocessing (SMP) systems where a single node may consist of hundreds of CPUs and massive amounts of memory.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier of this node. Note: this field is not user-assigned and is generated by the database.
<b>architecture</b>	<b>String</b>	No	This node's processor architecture.
<b>attributes</b>	<b>Map&lt;String, Map&gt;</b>	No	Attributes is a map of attribute names to tuples (maps) that describe the scheduling attributes of a node. Each tuple should contain the following entries: <ul style="list-style-type: none"> <li>• <b>value</b> - the attribute value</li> <li>• <b>displayValue</b> - the attribute display value</li> </ul>
<b>classes</b>	<b>Set&lt;String&gt;</b>	No	The classes that this node can be scheduled for.
<b>featuresCustom</b>	<b>Set&lt;String&gt;</b>	Yes	The features this node advertises which are customizable at run-time. This can be used to define node sets. (See also: <a href="#">featuresReported</a> .)

Field Name	Type	P UT	Description
<b>featuresReported</b>	<b>Set&lt;String&gt;</b>	No	The features this node advertises which are reported by resource managers or are present in the Moab Workload Manager configuration. This can be used to define node sets. (See also: <a href="#">featuresCustom</a> .)
<b>index</b>	<b>Integer</b>	No	The index for this node as reported by the resource manager.
<b>ipAddress</b>	<b>String</b>	No	This node's IPv4 address.
<b>isHypervisor</b>	<b>Boolean</b>	No	True if the node is a hypervisor, false otherwise. This is based on the <a href="#">NodeOperatingSystemInformation.hypervisorType</a> field. If hypervisorType is present, the node is a hypervisor. If it is null, then it is not a hypervisor.
<b>jobs</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Jobs associated with this node.
<b>lastUpdatedDate</b>	<b>Date</b>	No	The timestamp of the last moment when this node was updated. There is no guarantee that all user modifications to a node would be picked up. This will also be changed every RMPOLLINTERVAL even if a resource manager does not report information on this node.
<b>messages</b>	<a href="#">Set&lt;Message&gt;</a>	Yes	The list of messages attached to this node. They can be attached by admins, the resource manager layer, or triggers.

Field Name	Type	PUT	Description
<b>metrics</b>	<b>Map&lt;String, Double&gt;</b>	Yes	<p>Metrics are the measurable, quantitative, and changing aspects of this node. They are used to define workload placement, attach triggers, etc. There are some built-in metrics:</p> <ul style="list-style-type: none"> <li>• <b>speed</b> - A number from 0.0 to 1.0 describing the relative speed of the system for computational tasks. This is a composite metric, and is defined on a per-site basis.</li> <li>• <b>cpuLoad</b> - This is the CPU load on this node. This value is defined at the resource manager layer, but is generally defined on a per-operating system basis. For example, Unix-based OS's use some aspect of the Unix load average, as reported by the resource manager layer, while Windows-based OS's use CPU utilization.</li> </ul>
<b>migrationDisabled</b>	<b>Boolean</b>	No	True if VM migration is disabled on this node.
<b>name</b>	<b>String</b>	No	The name of this node. This name is unique <i>per instance</i> of Moab Workload Manager (i.e. not globally).
<b>operatingSystem</b>	<a href="#">NodeOperatingSystemInformation</a>	Yes	Describes the current or expected operating system image information for this node. The operatingSystem.image field can be changed using PUT.
<b>partition</b>	<b>String</b>	Yes	The partition this node belongs to.
<b>processorSpeed</b>	<b>Integer</b>	No	The speed, in MHz, or the processors on this node.

Field Name	Type	P U T	Description
<b>profilingEnabled</b>	<b>Boolean</b>	No	Indicates whether historical data gathering and reporting is enabled for this node. This is also controlled by the same setting on the default node (i.e. all nodes). If set to false (default), node statistics are not gathered.
<b>rack</b>	<b>Integer</b>	No	The rack where this node is located in the datacenter/cluster.
<b>reservations</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Reservations associated with this node.
<b>resourceManagerMessages</b>	<b>Map&lt;String, Map&gt;</b>	No	The resource manager messages for this node. Each key is the name of a resource manager, and the value is the message that the resource manager has posted onto the node.
<b>resourceManagers</b>	<a href="#">Set&lt;NodeResourceManager&gt;</a>	No	The resource managers that are reporting or have previously reported this node. Each object also contains information on the resource manager reports.
<b>resources</b>	<a href="#">Map&lt;String, Resource&gt;</a>	No	Contains references of a string representing a resource name to a resource object detailing the amount of the resource that is available, configured, etc. Each key is the name of the resource, which equates to the generic resource identifier or one of "processors", "memory", "disk", or "swap". This name may be used as an id in the resource types web service.
<b>slot</b>	<b>Integer</b>	No	The slot in the rack where this node is located.

Field Name	Type	PUT	Description
<b>states</b>	<a href="#">NodeStateInformation</a>	Yes	This node's state. The states.powerState and states.state fields can be changed using PUT.
<b>triggers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Triggers associated with this node.
<b>type</b>	<a href="#">NodeType</a>	No	The type of this node is governed by the types of resources it offers.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Variables is a map of key-value pairs, synonymous, but not directly related to, environment variables. They provide the mechanism to store arbitrary metadata which is useful to external systems in memory on this node.
<b>virtualContainers</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	The set of virtual containers that directly (not recursively) contain this node.
<b>virtualMachines</b>	<a href="#">Set&lt;DomainProxy&gt;</a>	No	Virtual machines associated with this node.

DomainProxy

A reference to an object contained within an object. For example, a Virtual Machine object contains a reference to the Node on which it is running. That reference is represented by this class.

Field Name	Type	PUT	Description
<b>name</b>	<b>String</b>	No	The name of the object.

Message

Field Name	Type	PUT	Description
<b>count</b>	<b>Integer</b>	No	The number of times this message has occurred.
<b>createdDate</b>	<b>Date</b>	No	The date this message was created.

Field Name	Type	PUT	Description
<b>expireDate</b>	<b>Date</b>	No	The date this message expires.
<b>message</b>	<b>String</b>	Yes	The message itself.

NodeOperatingSystemInformation

Describes the current or expected operating system image information for a node.

Field Name	Type	PUT	Description
<b>hypervisorType</b>	<b>String</b>	No	The hypervisor technology that this node uses. May be null if the node is not a hypervisor.
<b>image</b>	<b>String</b>	Yes	The name of the operating system currently running on this node. In cloud mode, this corresponds to the ID or name of an image in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>imageExpected</b>	<b>String</b>	No	The name of the image that was requested to run on this node (i.e. with <code>mnodectl -m os=myOs</code> ). In cloud mode, this corresponds to the ID or name of an image in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>imageLastUpdatedDate</b>	<b>Date</b>	No	The last time the image of this node was modified.
<b>imagesAvailable</b>	<b>Set&lt;String&gt;</b>	No	The list of image names that can be applied to this node. In cloud mode, this corresponds to IDs or names of images in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)
<b>virtualMachineImages</b>	<b>Set&lt;String&gt;</b>	No	The list of virtual machine image names the node is capable of supporting. In cloud mode, this corresponds to IDs or names of images in the image management API in MWS. (See also: <a href="#">Image.id</a> , <a href="#">Image.name</a> .)

NodeResourceManager

Field Name	Type	PUT	Description
<b>isMaster</b>	<b>Boolean</b>	No	Indicates whether this resource manager is the "master" of this Node. If true, it means that this resource manager has the final say on all properties reported about this Node. Note that the first resource manager to report a node is the master resource manager.
<b>name</b>	<b>String</b>	No	The name of the resource manager, according to Moab. This name appears in both the RMCFG parameter, and when diagnosing resource managers (e.g. <code>mdiag -R</code> ).
<b>stateReported</b>	<a href="#">NodeState</a>	No	The state reported by this resource manager. See the State section for more details.

### NodeState

This enumeration tracks the state of a node.

Value	Description
<b>NONE</b>	The node is set to none by the resource manager.
<b>DOWN</b>	The node is not available for workload.
<b>IDLE</b>	The node is available for workload but is not running anything.
<b>BUSY</b>	The node is running workload and cannot accept more.
<b>RUNNING</b>	The node is running workload and can accept more.
<b>DRAINED</b>	The node has been sent the drain request and has no workload on it.
<b>DRAINING</b>	The node has been sent the drain request, but still has workload on it.
<b>FLUSH</b>	The node is being reprovisioned.
<b>RESERVED</b>	The node is being reserved. This is an internal Moab state.
<b>UNKNOWN</b>	The state of the node is unknown.

Resource

Represents counts of resources available, configured, etc.

Field Name	Type	PUT	Description
<b>available</b>	<b>Integer</b>	No	The amount of this resource that is currently available for allocation to workload.
<b>configured</b>	<b>Integer</b>	No	The amount of this resource that is considered possible to schedule. Overcommit specifically applies to this, in other words, <code>configured = overcommitFactor * real</code> .
<b>dedicated</b>	<b>Integer</b>	No	The amount of this resource that has been allocated for running workload. When used in a job submission, this number is the amount of the resource required by the job.
<b>real</b>	<b>Integer</b>	No	The amount of this resource that physically exists on the node. Overcommit specifically doesn't apply to this. Note that overcommit currently only applies to "processors" and "memory", and so, for most cases, real and configured will always be the same.
<b>utilized</b>	<b>Integer</b>	No	The amount of this resource that is currently reported as utilized by resource managers.

NodeStateInformation

Field Name	Type	PUT	Description
<b>powerState</b>	<a href="#">NodePower</a>	Yes	The state of the node's power system, as reported by the RM layer. Modifying the powerState is possible, and, if Moab is configured properly, a request will be made to modify the power state accordingly.
<b>powerStateExpected</b>	<a href="#">NodePower</a>	No	The expected state of the node's power system. If a user has requested that a node be powered off (e.g. by modifying the <a href="#">powerState</a> attribute to <a href="#">NodePower.OFF</a> ), the requested state will be shown in this field until the state change is completed. If there is no pending power change request, this will be null.

Field Name	Type	PUT	Description
<b>state</b>	<a href="#">NodeState</a>	Yes	The scheduling state of the Node, as reported by the resource management layer.
<b>stateExpected</b>	<a href="#">NodeState</a>	No	The scheduling state of the Node, as expected by Moab. For example, Moab may think that a Node is "Busy" because it has allocated all configured resources, but a resource manager may report the state as "Running" based on actual utilization of the resources.
<b>stateLastUpdatedDate</b>	<b>Date</b>	No	A timestamp recording when the state of the Node was last modified.
<b>subState</b>	<b>String</b>	No	A text description of the state of the Node, with the intention of giving more details. Resource Managers may use this field to further describe the state being reported. Resource Managers should provide documented meaning to the possible sub-states that they can report.
<b>subStateLast</b>	<b>String</b>	No	The previous sub-state of the Node as reported by the resource management layer.
<b>subStateLastUpdatedDate</b>	<b>Date</b>	No	A timestamp recording when the sub-state was last modified.

### [NodePower](#)

Represents the various options for a Node's power state.

Value	Description
<b>NONE</b>	
<b>ON</b>	
<b>OFF</b>	

### [NodeType](#)

Represents the type of node as reported by a resource manager.

Value	Description
<b>Compute</b>	
<b>License</b>	
<b>Network</b>	
<b>Storage</b>	

## API version 1

NodeVersion1

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique ID of the node.
<b>accessPolicy</b>	<a href="#">NodeAccessPolicy</a>	No	The node's access policy.
<b>aliases</b>	<b>Set&lt;String&gt;</b>	No	Holds DNS aliases associated with the node
<b>architecture</b>	<b>String</b>	No	The node's processor architecture.
<b>availableClasses</b>	<b>Set&lt;String&gt;</b>	No	The classes that are available to the node.
<b>availableDisk</b>	<b>Integer</b>	No	The node's available disk space in MB.
<b>availableEndDate</b>	<b>Date</b>	No	The time when the node will stop being available.
<b>availableGenericResources</b>	<b>Map&lt;String, Integer&gt;</b>	No	The node's available generic resources. Stored as name and associated amount.
<b>availableMemory</b>	<b>Integer</b>	No	The node's available memory in MB.
<b>availableProcessors</b>	<b>Integer</b>	No	The node's available processors.
<b>availableStartDate</b>	<b>Date</b>	No	The time when the node will become available.
<b>availableSwap</b>	<b>Integer</b>	No	The node's available swap space in MB.

Field Name	Type	PUT	Description
<b>blockReason</b>	<b>String</b>	No	The reason why jobs are blocked from running on the node.
<b>comments</b>	<b>String</b>	No	User-specified comments regarding the node.
<b>configuredClasses</b>	<b>Set&lt;String&gt;</b>	No	The classes reported by the resource manager for the node.
<b>cpuLoad</b>	<b>Double</b>	No	The current load of the node as a percentage.
<b>dynamic</b>	<b>Boolean</b>	No	If true, the node is associated with a dynamic resource manager.
<b>externalLoad</b>	<b>Double</b>	No	The load on the node not attributed to any batch queues.
<b>features</b>	<b>Set&lt;String&gt;</b>	No	The collection of any opaque node features used to describe and categorize the node.
<b>flags</b>	<a href="#"><u>Set&lt;NodeFlag&gt;</u></a>	No	The flags that are set on this node. Valid only during GET.
<b>genericEvents</b>	<a href="#"><u>Set&lt;GenericEvent&gt;</u></a>	Yes	The list of generic events reported on this node.
<b>genericMetrics</b>	<b>Map&lt;String, Double&gt;</b>	Yes	The values for the generic metrics for this node.
<b>genericResources</b>	<b>Map&lt;String, Integer&gt;</b>	No	The generic resources attached to this node. With each generic resource there is a name and an associated amount.

Field Name	Type	PUT	Description
<b>hypervisorType</b>	<b>String</b>	No	The hypervisor type. Null if this is not a hypervisor.
<b>iOLoad</b>	<b>Double</b>	No	The load reported by the node on its I/O.
<b>index</b>	<b>Integer</b>	No	The index for this node as reported by the resource manager.
<b>jobs</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	The list of jobs on the node.
<b>lastStateUpdateDate</b>	<b>Date</b>	No	The date of the last resource manager update of the node's state.
<b>lastUpdateDate</b>	<b>Date</b>	No	The date of the last resource manager update
<b>maxIOIn</b>	<b>Double</b>	No	The maximum input as a percentage allowed for the node's I/O.
<b>maxIOLoad</b>	<b>Double</b>	No	The maximum total load as a percentage allowed for the node's I/O.
<b>maxIOOut</b>	<b>Double</b>	No	The maximum output as a percentage allowed for the node's I/O.
<b>maxJob</b>	<b>Integer</b>	No	The maximum number of jobs allowed to run on the node.
<b>maxJobPerUser</b>	<b>Integer</b>	No	The maximum number of jobs for a single user allowed to run on the node.

Field Name	Type	PUT	Description
<b>maxLoad</b>	<b>Double</b>	No	The maximum total load as a percentage allowed on the node.
<b>maxPEPerJob</b>	<b>Double</b>	No	The maximum processor equivalence per job allowed on the node.
<b>maxPageIn</b>	<b>Double</b>	No	The maximum total page in allowed on the node.
<b>maxPageOut</b>	<b>Double</b>	No	The maximum total page out allowed on the node.
<b>maxProc</b>	<b>Integer</b>	No	The maximum number of processors available on the node.
<b>maxProcPerClass</b>	<b>Integer</b>	No	The maximum number of processors per class available on the node.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	Yes	The list of system and user messages on the node.
<b>network</b>	<b>String</b>	No	The name of the node's network.
<b>networkAddress</b>	<b>String</b>	No	The network (usually IP) address of this node.
<b>networkLoad</b>	<b>Double</b>	No	The load of the node's network as a percentage.
<b>nextOS</b>	<b>String</b>	No	If populated, the next operating system that is currently being provisioned.
<b>operations</b>	<a href="#">Set&lt;NodeOperation&gt;</a>	No	The list of operations that can be performed on this node.

Field Name	Type	PUT	Description
<b>os</b>	<b>String</b>	Yes	The operating system of the node.
<b>osList</b>	<b>Set&lt;String&gt;</b>	No	The list of potential operating systems for the node.
<b>overcommit</b>	<a href="#"><u>OverCommitFactor</u></a>	No	This is the coefficient that determines how much resources may be overutilized on this hypervisor. Will normally be null for non-hypervisor nodes.
<b>partition</b>	<b>String</b>	Yes	The partition this node belongs to.
<b>power</b>	<a href="#"><u>NodePower</u></a>	Yes	The node's power state according to the resource managers reporting this node.
<b>powerPolicy</b>	<a href="#"><u>PowerPolicy</u></a>	No	The power policy for this node. The default is static where the power is controlled manually.
<b>powerSelected</b>	<a href="#"><u>NodePower</u></a>	No	The node's power state that Moab is currently scheduling. This could be different from the actual node power itself.
<b>priority</b>	<b>Double</b>	No	The priority associated with the node.
<b>priorityFunction</b>	<b>String</b>	No	The function used to calculate this node's priority.
<b>procSpeed</b>	<b>Integer</b>	No	The node's processor speed in MHz.

Field Name	Type	PUT	Description
<b>profilingEnabled</b>	<b>Boolean</b>	No	Indicates whether statistical profiling is enabled for this node. If set to false, node statistics will not be gathered. Defaults to false.
<b>rack</b>	<b>Integer</b>	No	The node's rack location.
<b>reservationCount</b>	<b>Integer</b>	No	The number of reservations on the node.
<b>reservations</b>	<b>Set&lt;String&gt;</b>	No	The list of reservations' names currently on this node.
<b>rmAccessList</b>	<b>String</b>	No	A comma-separated list of resource managers that have access to this node.
<b>size</b>	<b>Integer</b>	No	The number of slots or size units consumed by the node.
<b>slot</b>	<b>Integer</b>	No	The node's slot location.
<b>speed</b>	<b>Double</b>	No	The relative speed of the node. The speed must be between .1 and 100.0. The default value for speed is 1.0.
<b>speedWeight</b>	<b>Double</b>	No	The weight of the speed metric in the node's priority function
<b>state</b>	<a href="#"><u>NodeState</u></a>	Yes	The node's current state.
<b>substate</b>	<b>String</b>	No	The substate that will give more information about the status of the VM.
<b>taskCount</b>	<b>Integer</b>	No	The number of tasks currently running on the node.

Field Name	Type	PUT	Description
<b>totalActiveTime</b>	<b>Long</b>	No	The epoch time in seconds that the node was tracked as being busy with jobs.
<b>totalAvailableTime</b>	<b>Long</b>	No	The epoch time in seconds that the node is available.
<b>totalDisk</b>	<b>Integer</b>	No	The node's total disk space in MB.
<b>totalMemory</b>	<b>Integer</b>	No	The node's total memory in MB.
<b>totalProcessors</b>	<b>Integer</b>	No	The node's total processors.
<b>totalStatsTime</b>	<b>Long</b>	No	The epoch time in seconds that stats have been tracked for the node.
<b>totalSwap</b>	<b>Integer</b>	No	The node's total swap space in MB.
<b>totalUpTime</b>	<b>Long</b>	No	The epoch time in seconds that the node was tracked as being up.
<b>type</b>	<b>String</b>	No	The type of the node. This is used for allocation management purposes.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Variables associated with the node.
<b>virtualMachines</b>	<a href="#">Set&lt;VirtualMachineVersion1&gt;</a>	No	The list of virtual machines on the node.
<b>vmOsList</b>	<b>Set&lt;String&gt;</b>	No	The list of potential operating systems that can be used when creating a virtual machine on the node

### NodeAccessPolicy

This enumeration describes how node resources will be shared by various tasks.

Value	Description
<b>NONE</b>	
<b>SHARED</b>	Tasks from any combination of jobs may utilize available resources.
<b>SHAREDONLY</b>	Only jobs requesting shared node access may utilize available resources.
<b>SINGLEJOB</b>	Tasks from a single job may utilize available resources.
<b>SINGLETASK</b>	A single task from a single job may run on the node.
<b>SINGLEUSER</b>	Tasks from any jobs owned by the same user may utilize available resources.
<b>UNIQUEUSER</b>	Any number of tasks from a single job may allocate resources from a node but only if the user has no other jobs running on that node.
<b>SINGLEGROUP</b>	Any number of tasks from the same group may utilize node.
<b>SINGLEACCOUNT</b>	Any number of tasks from the same account may utilize node.

### NodeFlag

Represents the various flags for nodes.

Value	Description
<b>NONE</b>	
<b>GLOBAL_VARS</b>	Node has variables available for triggers.
<b>NO_VM_MIGRATIONS</b>	Node is excluded from all virtual machine migration decisions.

### GenericEvent

A `GenericEvent` is an event that is reported by a [NodeResourceManager](#) on a [Node](#) or [VirtualMachine](#). Usually, there is some corresponding action defined in the `moab.cfg` file.

GenericEvents are defined by a "name" characteristic, which indicates the type of thing happened on the resource associated with it. For example, a "highcputemp" GenericEvent might be reported by a NodeResourceManager when it detects that the CPU temperature has risen about 50 degrees C.

Note that GenericEvents vary by configuration: one installation of Moab may report GenericEvents of "highcputemp" while another would report "cputemphigh". Although the *meaning* is the same, the *type* or *name* is different.

Field Name	Type	PUT	Description
<b>date</b>	<b>Date</b>	No	The timestamp of when this GenericEvent was <i>reported</i> to Moab
<b>message</b>	<b>String</b>	Yes	The message that goes with this GenericEvent
<b>name</b>	<b>String</b>	Yes	The name of the GenericEvent. In other words, the type.
<b>severity</b>	<b>Integer</b>	No	The severity level of the event (1-4, 1 is low severity, 0 is unset)

DomainProxyVersion1

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The id of the object.

MessageVersion1

Field Name	Type	PUT	Description
<b>author</b>	<b>String</b>	No	The author of the message.
<b>creationTime</b>	<b>Date</b>	No	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	No	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	No	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	Yes	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	No	The number of times this message has been displayed.

Field Name	Type	PUT	Description
<b>priority</b>	<b>Double</b>	No	An optional priority that can be attached to the comment.

### NodeOperation

This class represents an operation that can be taken on a given node.

Value	Description
<b>CREATE_VM</b>	
<b>MIGRATE_VM</b>	
<b>MODIFY_OS</b>	
<b>OFFLINE</b>	
<b>ONLINE</b>	
<b>POWER_ON</b>	
<b>POWER_OFF</b>	
<b>REBOOT</b>	

### OverCommitFactor

This class represents the multipliers to use for over committing the basic grouping of resource types that Moab generally uses when scheduling and taking policy actions in a system. These resource types are the following:

- Disk
- Memory
- Processors
- Swap

These over commit factors are used in multiplying the reported resources from an RM for servers. In addition, these factors are used in establishing when VMs are migrated away from a hypervisor.

Field Name	Type	PUT	Description
<b>overcommitMap</b>	<b>Map&lt;String, Double&gt;</b>	No	

NodePower

Represents the various options for a Node's power state.

Value	Description
<b>NONE</b>	
<b>ON</b>	
<b>OFF</b>	

PowerPolicy

This enumeration represents the power policy to take when deciding about resource power.

Value	Description
<b>NONE</b>	
<b>ONDEMAND</b>	Moab will attempt to power the server off if there is no workload running on it.
<b>STATIC</b>	The power is handled manually.
<b>GREEN</b>	

NodeState

This enumeration tracks the state of a node.

Value	Description
<b>NONE</b>	The node is set to none by the resource manager.

Value	Description
<b>DOWN</b>	The node is not available for workload.
<b>IDLE</b>	The node is available for workload but is not running anything.
<b>BUSY</b>	The node is running workload and cannot accept more.
<b>RUNNING</b>	The node is running workload and can accept more.
<b>DRAINED</b>	The node has been sent the drain request and has no workload on it.
<b>DRAINING</b>	The node has been sent the drain request, but still has workload on it.
<b>FLUSH</b>	The node is being reprovisioned.
<b>RESERVED</b>	The node is being reserved. This is an internal Moab state.
<b>UNKNOWN</b>	The state of the node is unknown.

#### VirtualMachineVersion1

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique ID of the VM.
<b>aliases</b>	<b>Set&lt;String&gt;</b>	No	The list of aliases that refer to this VM.
<b>availableDisk</b>	<b>Integer</b>	No	The amount of disk space available on this VM.
<b>availableMemory</b>	<b>Integer</b>	No	The amount of memory available on this VM.
<b>availableProcessors</b>	<b>Integer</b>	No	The number of processors available on this VM.
<b>cpuLoad</b>	<b>Double</b>	No	The CPU load for this VM.

Field Name	Type	PUT	Description
<b>description</b>	<b>String</b>	No	A description of this VM.
<b>effectiveTimeToLive</b>	<b>Long</b>	No	Time (in seconds) that this VM has to run (0 means not set).
<b>flags</b>	<a href="#"><u>Set&lt;VMFlag&gt;</u></a>	Yes	The flags associated with this VM.
<b>genericEvents</b>	<a href="#"><u>Set&lt;GenericEvent&gt;</u></a>	Yes	The list of generic events reported on this VM.
<b>genericMetrics</b>	<b>Map&lt;String, Double&gt;</b>	Yes	The values for the generic metrics for this VM.
<b>lastMigrationDate</b>	<b>Date</b>	No	The date when this VM was last migrated.
<b>lastSubstate</b>	<b>String</b>	No	Last reported substate (substate is not always reported).
<b>lastSubstateModificationDate</b>	<b>Date</b>	No	Date/time that lastSubstate was modified.
<b>lastUpdateDate</b>	<b>Date</b>	No	The date of when this VM was last updated.
<b>migrationCount</b>	<b>Integer</b>	No	The number of times this VM has been migrated.
<b>networkAddress</b>	<b>String</b>	No	The network address of this VM.
<b>node</b>	<a href="#"><u>DomainProxyVersion1</u></a>	Yes	The physical node on which this VM is running.
<b>os</b>	<b>String</b>	Yes	The current operating system of the VM.
<b>osList</b>	<b>Set&lt;String&gt;</b>	No	The list of operating systems that the VM can run.

Field Name	Type	PUT	Description
<b>powerSelectState</b>	<a href="#">NodePower</a>	No	The selected power state of this VM.
<b>powerState</b>	<a href="#">NodePower</a>	Yes	The power state of this VM.
<b>rack</b>	<b>Integer</b>	No	The rack of the VM inside a given node.
<b>requestedTimeToLive</b>	<b>Long</b>	No	User-specified time (in seconds) that this VM has to run (0 means not set).
<b>slot</b>	<b>Integer</b>	No	The slot of the VM inside a given node.
<b>sovereign</b>	<b>Boolean</b>	No	
<b>startDate</b>	<b>Date</b>	No	Date/time that this VM was started (ready and usable)
<b>state</b>	<a href="#">NodeState</a>	Yes	The state of the VM.
<b>storage</b>	<b>String</b>	No	
<b>substate</b>	<b>String</b>	No	The substate that will give more information about the status of the VM.
<b>template</b>	<b>String</b>	No	
<b>totalDisk</b>	<b>Integer</b>	No	The amount of disk space configured in MB.
<b>totalMemory</b>	<b>Integer</b>	No	The amount of memory configured in MB.
<b>totalProcessors</b>	<b>Integer</b>	No	The number of virtual processors in the VM.
<b>trackingJob</b>	<a href="#">DomainProxyVersion1</a>	No	The tracking job for this VM.

Field Name	Type	PUT	Description
<b>triggers</b>	<a href="#">Set&lt;Trigger&gt;</a>	Yes	The list of triggers on this VM.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Variables associated with this VM.

VMFlag

This enumeration specifies the VM flag types of a virtual machine.

Value	Description
<b>CAN_MIGRATE</b>	VM may be directly migrated without direction from parent workload.
<b>CANNOT_MIGRATE</b>	This value is used to remove the CAN_MIGRATE flag.
<b>CONSUMER</b>	VM should be treated as resource consumer regardless of Moab-tracked workload.
<b>CREATION_COMPLETED</b>	VM creation completed.
<b>DELETED</b>	VM deleted.
<b>DESTROY_PENDING</b>	VM destroy requested.
<b>DESTROYED</b>	VM reported destroyed by Resource Manager (VM will be removed when resource manager no longer reports VM).
<b>INITIALIZING</b>	VM is initializing.
<b>ONETIMEUSE</b>	VM is was created for a job with vmusagepolicy=createvm.
<b>SHARED</b>	Multiple jobs can use VM resources simultaneously.
<b>UTILIZATION_REPORTED</b>	

Trigger

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	No	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	No	
<b>blockTime</b>	<b>Date</b>	No	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	No	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	No	
<b>expireTime</b>	<b>Date</b>	No	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	<b>Date</b>	No	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	No	
<b>interval</b>	<b>Boolean</b>	No	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	No	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	No	Specifies whether this trigger can fire multiple times. Defaults to false.

Field Name	Type	PUT	Description
<b>name</b>	<b>String</b>	No	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	No	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	No	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	No	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#">TriggerPeriod</a>	No	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	No	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	No	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	No	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	No	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.

Field Name	Type	PUT	Description
<b>timeout</b>	<b>Date</b>	No	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	No	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	

Value	Description
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id

Value	Description
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	<p>By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).</p> <p>NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.</p>

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
MINUTE	
HOUR	
DAY	
WEEK	
MONTH	

Related topics

- [Nodes on page 1573](#)

Fields: Notification Conditions

 See the associated [Notification conditions on page 1580](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	notification-conditions	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	notification-conditions.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

### NotificationCondition

A notification condition is related to an [Event](#), but differs in three distinct areas:

- Notification conditions are a persistent condition of the system or a component rather than a single occurrence.
  - They are ongoing rather than reoccurring, which is why they are generated from [NotificationConditions](#).
  - They may be observed many times, but the condition is always the same.
  - A good test for this is if something "is" wrong rather than something "went" wrong.
- Notification conditions can be acted on to result in a resolved state, mean the administrator or user can and must take actions to "fix" the condition or problem.
- Notification conditions contain state information based on administrator or user input, meaning that they contain information about the condition (similar to events), but also contain the "status" of the administrator's view of the notification, whether it is currently open, dismissed, or ignored.

In general, questions may be asked to ascertain whether an Event or a Notification Condition is the right fit for an occurrence. These questions, along with some sample situations, are provided below.

- Is the occurrence the root cause of a potentially ongoing condition?
  - A VM migration failed because the VM's state was unknown. The root cause was that the state was unknown, not that the VM migration failed. Therefore, VM migration failed would be an event, while the unknown state would be a notification condition.
  - A VM service provision fails because there are no hypervisors that satisfy the requirements. This would be an event. Note that there may be a notification related to this failure, such as a service template requires a feature that does not exist on *any* hypervisors in the system, but this would be distinctly detected and managed from the provision failure event.

- A request to MWS failed because the connection between MWM and MongoDB was misconfigured. The failed request may be represented as an event, but a notification condition should exist that the connection between MWM and MongoDB was down.
- Can an administrator or user affect the outcome of the occurrence?
    - The outcome of a VM migration failing is in the past and cannot be changed by the administrator. However, the outcome of a *future* VM migration may be changed when the administrator resolves the root problem (i.e. VM state is unknown).

A notification condition is an observed condition for which [Notifications](#) are created. These conditions are created or updated on every PUT request based on the [NotificationCondition.escalationLevel](#), [NotificationCondition.origin](#), [NotificationCondition.message](#), [NotificationCondition.objectType](#), and [NotificationCondition.objectId](#) fields. When notifications are requested, these observed conditions are used to create the notifications for the requesting user.

While notification conditions may not be deleted, they "expire" after a specified amount of time and are no longer considered as active conditions for which notifications are created.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The identifier of the condition.
<b>createdDate</b>	<b>Date</b>	No	The date that the condition first started appearing.
<b>details</b>	<b>Map&lt;String, Map&gt;</b>	No	Arbitrary storage of details for this notification. This could include "pluginType", "pluginId", etc.
<b>escalationLevel</b>	<a href="#">EscalationLevel</a>	No	The escalation level of the condition. This indicates who should care about the condition or who can respond to it. This may NOT be <a href="#">EscalationLevel.INTERNAL</a> .
<b>expirationDate</b>	<b>Date</b>	No	The date at which the condition is considered "expired" and notifications are no longer created for it. This is typically set using the <a href="#">expirationDuration</a> field.

Field Name	Type	PUT	Description
<b>expirationDuration</b>	<b>Long</b>	No	The duration in seconds that may pass before a notification will not be created for a user. Effectively this can disable notifications from being created if they are too old. When this field is set, it will set the <a href="#">expirationDate</a> field automatically each time the condition is updated or on creation. This field must be set to 1 or greater or else set to null.
<b>message</b>	<b>String</b>	No	A message detailing the notification and why it exists, with possible action items.
<b>objectId</b>	<b>String</b>	No	The identifier of the object which this notification affects, such as "node1" or "vm1".
<b>objectType</b>	<b>String</b>	No	The object type that this notification affects, such as "Node", "VM", "System", etc.
<b>observedDate</b>	<b>Date</b>	No	The latest date that the condition was observed. If this field is not set in an update request, it will automatically be set to the current date.
<b>origin</b>	<b>String</b>	No	The origin of the notification.
<b>tenant</b>	<b>Map&lt;String, Map&gt;</b>	No	The tenant that this notification came from. (contains tenant id and name)

EscalationLevel

Value	Description
<b>USER</b>	
<b>POWER_USER</b>	
<b>ADMIN</b>	
<b>INTERNAL</b>	

Related topics

- [Notification conditions on page 1580](#)

## Fields: Notifications

**i** See the associated [Notifications on page 1585](#) resource section for more information on how to use this resource and supported operations.

### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	notifications	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	notifications.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

Notification

Notifications, while related to [Events](#), are used for different purposes. See [NotificationCondition](#) for more information on when notifications should be used as opposed to events.

Notifications are a per-user representation of all notification conditions present in the system at any one time. When an administrator or user requests this resource, notifications are automatically created from the notification conditions that they have access to (determined by the [Notification.tenant](#) or the [NotificationCondition.escalationLevel](#) fields).

Notifications are expected to contain messages and details that may be understood by a user or admin depending on the escalation level, and contain fields that control whether the user or admin will be notified of future updates to their corresponding condition.

Notifications cannot be deleted, but they can be marked as ignored (see [Notification.ignoredDate](#) or dismissed (see [Notification.dismissedDate](#)).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The identifier of the notification.
<b>conditionId</b>	<b>String</b>	No	The identifier of the <a href="#">NotificationCondition</a> from which this notification was created.
<b>createdDate</b>	<b>Date</b>	No	The date that the notification condition first appeared.
<b>details</b>	<b>Map&lt;String, Map&gt;</b>	No	Arbitrary storage of details for this notification. This could include "pluginType", "pluginId", etc.
<b>dismissedDate</b>	<b>Date</b>	No	The date that the notification was dismissed by a user or admin, meaning that they acknowledged the notification and wanted to know of future updates to this notification. This field is cleared every time the attached notification condition is updated/observed again. (See also: <a href="#">conditionId</a> .)
<b>ignoredDate</b>	<b>Date</b>	No	The date that the notification was ignored by a user or admin, meaning that they acknowledged the notification now and in the future and did not wish to know of any updates. This field is never cleared, even if the attached notification condition is updated/observed again.
<b>message</b>	<b>String</b>	No	A message detailing the notification and why it exists, with possible action items.

Field Name	Type	PUT	Description
<b>objectId</b>	<b>String</b>	No	The identifier of the object which this notification affects, such as "node1" or "vm1".
<b>objectType</b>	<b>String</b>	No	The object type that this notification affects, such as "Node", "VM", "System", etc.
<b>observedDate</b>	<b>Date</b>	No	The latest date that the notification condition was observed. If this field, <a href="#">ignoredDate</a> , and <a href="#">dismissedDate</a> are not set during an update (i.e. a user/admin is not ignoring or dismissing the notification), this field will automatically be set to the current date.
<b>origin</b>	<b>String</b>	No	The origin of the notification.
<b>user</b>	<b>String</b>	No	The user that this notification was created for.

Related topics

- [Notifications on page 1585](#)

## Fields: Plugins

 See the associated [Plugins on page 1599](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	plugins	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	plugins.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

PluginInstance

This class represents a configured plugin created from a plugin type.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	Yes	No	Unique identifier for the plugin. Must contain at least one letter and must also start with a letter. Reserved IDs are "all" and "moab". If these are used an error will be returned.
<b>autoStart</b>	<b>Boolean</b>	Yes	Yes	Whether the plugin should start automatically when created.
<b>config</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The configuration of the plugin. Plugin types may define constraints on the configuration, therefore it is recommended to view the plugin type's documentation for more information on required and optional fields. Regardless, the plugin configuration supports arbitrary keys and values.
<b>dateCreated</b>	<b>Date</b>	No	No	The date that this plugin was created.
<b>lastPollDate</b>	<b>Date</b>	No	No	The date of the last polling event that occurred. This may be null if the plugin is in the STOPPED state or has not yet been polled.
<b>lastUpdated</b>	<b>Date</b>	No	No	The date that this plugin was last updated.
<b>nextPollDate</b>	<b>Date</b>	No	No	The date of the next polling event that is scheduled to occur. This may be null if the plugin is in the STOPPED state.
<b>pluginType</b>	<b>String</b>	Yes	No	The plugin name as in Native or Example for the plugin called ExamplePlugin.
<b>pollInterval</b>	<b>Integer</b>	Yes	Yes	The polling interval to use for the plugin in seconds. This is ignored if the plugin type does not support polling.

Field Name	Type	POST	PUT	Description
<b>precedence</b>	<b>Long</b>	Yes	Yes	<p>The precedence of this plugin, with the lowest value being the highest precedence. Minimum of 1. This is used when doing data consolidation when reporting current state data. Lower numbers results in a higher precedence (i.e. 1 is higher precedence than 10).</p> <p>If not specified during creation, this will be automatically set to 1 for the first plugin created, then 1 greater for each subsequently created plugin (i.e. 1 for plugin1, 2 for plugin2, etc). It is always set to 1 greater than the plugin with the greatest precedence number (i.e. 11 if two plugins exist with precedence 1 and 10).</p>
<b>state</b>	<a href="#"><u>PluginState</u></a>	No	No	The current state of the plugin. Defaults to <a href="#"><u>PluginState.STOPPED</u></a> .

[PluginState](#)

Represents the current state of a plugin.

Value	Description
<b>STOPPED</b>	The plugin is created and ready for use, but is not currently receiving any events
<b>STARTED</b>	The plugin is currently receiving events and is working correctly.
<b>PAUSED</b>	<p>The plugin is currently not receiving any events but is also not stopped.</p> <p>This should be used when polling or other events should stop only temporarily without firing the stop events.</p>
<b>ERRORED</b>	<p>MWS has detected an error with the plugin and has automatically stopped it. Errors could be due to the following reasons:</p> <ol style="list-style-type: none"> <li>1. An invalid configuration was detected when running the <a href="#"><u>AbstractPlugin.configure</u></a> method.</li> <li>2. An unexpected exception was thrown during an event, such as during polling.</li> </ol>

## API version 2

PluginInstance

This class represents a configured plugin created from a plugin type.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	Yes	No	Unique identifier for the plugin. Must contain at least one letter and must also start with a letter. Reserved IDs are "all" and "moab". If these are used an error will be returned.
<b>autoStart</b>	<b>Boolean</b>	Yes	Yes	Whether the plugin should start automatically when created.
<b>config</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The configuration of the plugin. Plugin types may define constraints on the configuration, therefore it is recommended to view the plugin type's documentation for more information on required and optional fields. Regardless, the plugin configuration supports arbitrary keys and values.
<b>dateCreated</b>	<b>Date</b>	No	No	The date that this plugin was created.
<b>lastPollDate</b>	<b>Date</b>	No	No	The date of the last polling event that occurred. This may be null if the plugin is in the STOPPED state or has not yet been polled.
<b>lastUpdated</b>	<b>Date</b>	No	No	The date that this plugin was last updated.
<b>nextPollDate</b>	<b>Date</b>	No	No	The date of the next polling event that is scheduled to occur. This may be null if the plugin is in the STOPPED state.
<b>pluginType</b>	<b>String</b>	Yes	No	The plugin name as in Native or Example for the plugin called ExamplePlugin.
<b>pollInterval</b>	<b>Integer</b>	Yes	Yes	The polling interval to use for the plugin in seconds. This is ignored if the plugin type does not support polling.

Field Name	Type	POST	PUT	Description
<b>precedence</b>	<b>Long</b>	Yes	Yes	<p>The precedence of this plugin, with the lowest value being the highest precedence. Minimum of 1. This is used when doing data consolidation when reporting current state data. Lower numbers results in a higher precedence (i.e. 1 is higher precedence than 10).</p> <p>If not specified during creation, this will be automatically set to 1 for the first plugin created, then 1 greater for each subsequently created plugin (i.e. 1 for plugin1, 2 for plugin2, etc). It is always set to 1 greater than the plugin with the greatest precedence number (i.e. 11 if two plugins exist with precedence 1 and 10).</p>
<b>state</b>	<a href="#"><u>PluginState</u></a>	No	No	The current state of the plugin. Defaults to <a href="#"><u>PluginState.STOPPED</u></a> .

[PluginState](#)

Represents the current state of a plugin.

Value	Description
<b>STOPPED</b>	The plugin is created and ready for use, but is not currently receiving any events
<b>STARTED</b>	The plugin is currently receiving events and is working correctly.
<b>PAUSED</b>	<p>The plugin is currently not receiving any events but is also not stopped.</p> <p>This should be used when polling or other events should stop only temporarily without firing the stop events.</p>
<b>ERRORED</b>	<p>MWS has detected an error with the plugin and has automatically stopped it. Errors could be due to the following reasons:</p> <ol style="list-style-type: none"> <li>1. An invalid configuration was detected when running the <a href="#"><u>AbstractPlugin.configure</u></a> method.</li> <li>2. An unexpected exception was thrown during an event, such as during polling.</li> </ol>

## API version 1

PluginInstance

This class represents a configured plugin created from a plugin type.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	Yes	No	Unique identifier for the plugin. Must contain at least one letter and must also start with a letter. Reserved IDs are "all" and "moab". If these are used an error will be returned.
<b>autoStart</b>	<b>Boolean</b>	Yes	Yes	Whether the plugin should start automatically when created.
<b>config</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The configuration of the plugin. Plugin types may define constraints on the configuration, therefore it is recommended to view the plugin type's documentation for more information on required and optional fields. Regardless, the plugin configuration supports arbitrary keys and values.
<b>dateCreated</b>	<b>Date</b>	No	No	The date that this plugin was created.
<b>lastPollDate</b>	<b>Date</b>	No	No	The date of the last polling event that occurred. This may be null if the plugin is in the STOPPED state or has not yet been polled.
<b>lastUpdated</b>	<b>Date</b>	No	No	The date that this plugin was last updated.
<b>nextPollDate</b>	<b>Date</b>	No	No	The date of the next polling event that is scheduled to occur. This may be null if the plugin is in the STOPPED state.
<b>pluginType</b>	<b>String</b>	Yes	No	The plugin name as in Native or Example for the plugin called ExamplePlugin.
<b>pollInterval</b>	<b>Integer</b>	Yes	Yes	The polling interval to use for the plugin in seconds. This is ignored if the plugin type does not support polling.

Field Name	Type	POST	PUT	Description
<b>precedence</b>	<b>Long</b>	Yes	Yes	The precedence of this plugin, with the lowest value being the highest precedence. Minimum of 1. This is used when doing data consolidation when reporting current state data. Lower numbers results in a higher precedence (i.e. 1 is higher precedence than 10). If not specified during creation, this will be automatically set to 1 for the first plugin created, then 1 greater for each subsequently created plugin (i.e. 1 for plugin1, 2 for plugin2, etc). It is always set to 1 greater than the plugin with the greatest precedence number (i.e. 11 if two plugins exist with precedence 1 and 10).
<b>state</b>	<a href="#"><u>PluginState</u></a>	No	No	The current state of the plugin. Defaults to <a href="#"><u>PluginState.STOPPED</u></a> .

[PluginState](#)

Represents the current state of a plugin.

Value	Description
<b>STOPPED</b>	The plugin is created and ready for use, but is not currently receiving any events
<b>STARTED</b>	The plugin is currently receiving events and is working correctly.
<b>PAUSED</b>	The plugin is currently not receiving any events but is also not stopped. This should be used when polling or other events should stop only temporarily without firing the stop events.
<b>ERRORED</b>	MWS has detected an error with the plugin and has automatically stopped it. Errors could be due to the following reasons:  <ol style="list-style-type: none"> <li>1. An invalid configuration was detected when running the <a href="#"><u>AbstractPlugin.configure</u></a> method.</li> <li>2. An unexpected exception was thrown during an event, such as during polling.</li> </ol>

Related topics

- [Plugins on page 1599](#)

## Fields: Plugin Types

**i** See the associated [Plugin types on page 1607](#) resource section for more information on how to use this resource and supported operations.

### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	plugin-types	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	plugin-types.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

PluginType

Represents a MWS plugin type. All fields in this class are generated from plugin project and type metadata and cannot be modified directly. Consequentially, all fields are only valid for list/show/GET operations.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier of the plugin type. This is based on the class name of the plugin. Ex:  <b>Plugin Class Name -&gt; ID</b> NativePlugin -> Native MSMPlugin -> MSM MyExamplePlugin -> MyExample
<b>author</b>	<b>String</b>	No	The main author (company or person) of the plugin type.
<b>commonsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of the plugin framework (plugins-commons dependency) is required for the plugin type. In the format 'COMMONS_VERSION > *', meaning that any version greater or equal to COMMONS_VERSION is valid.
<b>description</b>	<b>String</b>	No	The full description of the plugin type.
<b>documentationLink</b>	<b>String</b>	No	A full URL to the complete documentation for the plugin type.
<b>email</b>	<b>String</b>	No	The email of the author.
<b>eventComponent</b>	<b>Integer</b>	No	The event component ID of the plugin type. This should be unique for each plugin type and should be 1 or greater.
<b>initialPlugins</b>	<b>Map&lt;String, Map&gt;</b>	No	Represents the plugins that are initially configured when the plugin type is loaded. Each key represents the plugin ID.
<b>issueManagementLink</b>	<b>String</b>	No	A full URL to the issue management system or project for the plugin type.

Field Name	Type	PUT	Description
<b>license</b>	<b>String</b>	No	The license of this plugin type, typically APACHE.
<b>mwsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of MWS is required for the plugin type. In the format 'MWS_VERSION > *', meaning that any version greater or equal to MWS_VERSION is valid.
<b>pollMethod</b>	<b>boolean</b>	No	Indicates whether the plugin type has a defined 'poll' method (event handler) or not.
<b>realizedEventComponent</b>	<b>Integer</b>	No	The fully realized event component ID of the plugin type, including the MWS bits. This should take the form of 0x201. If the <a href="#">eventComponent</a> is not set, this will be 0x2FF, meaning the component ID is an unknown plugin type.
<b>scmLink</b>	<b>String</b>	No	A full URL to the Source Control Management (SCM) system or project for the plugin type.
<b>title</b>	<b>String</b>	No	A short name describing the plugin type.
<b>website</b>	<b>String</b>	No	The website of the author.

## API version 2

PluginType

Represents a MWS plugin type. All fields in this class are generated from plugin project and type metadata and cannot be modified directly. Consequentially, all fields are only valid for list/show/GET operations.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier of the plugin type. This is based on the class name of the plugin. Ex:  <b>Plugin Class Name -&gt; ID</b> NativePlugin -> Native MSMPlugin -> MSM MyExamplePlugin -> MyExample
<b>author</b>	<b>String</b>	No	The main author (company or person) of the plugin type.
<b>commonsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of the plugin framework (plugins-commons dependency) is required for the plugin type. In the format 'COMMONS_VERSION > *', meaning that any version greater or equal to COMMONS_VERSION is valid.
<b>description</b>	<b>String</b>	No	The full description of the plugin type.
<b>documentationLink</b>	<b>String</b>	No	A full URL to the complete documentation for the plugin type.
<b>email</b>	<b>String</b>	No	The email of the author.
<b>eventComponent</b>	<b>Integer</b>	No	The event component ID of the plugin type. This should be unique for each plugin type and should be 1 or greater.
<b>initialPlugins</b>	<b>Map&lt;String, Map&gt;</b>	No	Represents the plugins that are initially configured when the plugin type is loaded. Each key represents the plugin ID.
<b>issueManagementLink</b>	<b>String</b>	No	A full URL to the issue management system or project for the plugin type.

Field Name	Type	PUT	Description
<b>license</b>	<b>String</b>	No	The license of this plugin type, typically APACHE.
<b>mwsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of MWS is required for the plugin type. In the format 'MWS_VERSION > *', meaning that any version greater or equal to MWS_VERSION is valid.
<b>pollMethod</b>	<b>boolean</b>	No	Indicates whether the plugin type has a defined 'poll' method (event handler) or not.
<b>realizedEventComponent</b>	<b>Integer</b>	No	The fully realized event component ID of the plugin type, including the MWS bits. This should take the form of 0x201. If the <a href="#">eventComponent</a> is not set, this will be 0x2FF, meaning the component ID is an unknown plugin type.
<b>scmLink</b>	<b>String</b>	No	A full URL to the Source Control Management (SCM) system or project for the plugin type.
<b>title</b>	<b>String</b>	No	A short name describing the plugin type.
<b>website</b>	<b>String</b>	No	The website of the author.

## API version 1

PluginType

Represents a MWS plugin type. All fields in this class are generated from plugin project and type metadata and cannot be modified directly. Consequentially, all fields are only valid for list/show/GET operations.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier of the plugin type. This is based on the class name of the plugin. Ex:  <b>Plugin Class Name -&gt; ID</b> NativePlugin -> Native MSMPlugin -> MSM MyExamplePlugin -> MyExample
<b>author</b>	<b>String</b>	No	The main author (company or person) of the plugin type.
<b>commonsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of the plugin framework (plugins-commons dependency) is required for the plugin type. In the format 'COMMONS_VERSION > *', meaning that any version greater or equal to COMMONS_VERSION is valid.
<b>description</b>	<b>String</b>	No	The full description of the plugin type.
<b>documentationLink</b>	<b>String</b>	No	A full URL to the complete documentation for the plugin type.
<b>email</b>	<b>String</b>	No	The email of the author.
<b>eventComponent</b>	<b>Integer</b>	No	The event component ID of the plugin type. This should be unique for each plugin type and should be 1 or greater.
<b>initialPlugins</b>	<b>Map&lt;String, Map&gt;</b>	No	Represents the plugins that are initially configured when the plugin type is loaded. Each key represents the plugin ID.
<b>issueManagementLink</b>	<b>String</b>	No	A full URL to the issue management system or project for the plugin type.

Field Name	Type	PUT	Description
<b>license</b>	<b>String</b>	No	The license of this plugin type, typically APACHE.
<b>mwsVersion</b>	<b>String</b>	No	A string representing the restriction on which version of MWS is required for the plugin type. In the format 'MWS_VERSION > *', meaning that any version greater or equal to MWS_VERSION is valid.
<b>pollMethod</b>	<b>boolean</b>	No	Indicates whether the plugin type has a defined 'poll' method (event handler) or not.
<b>realizedEventComponent</b>	<b>Integer</b>	No	The fully realized event component ID of the plugin type, including the MWS bits. This should take the form of 0x201. If the <a href="#">eventComponent</a> is not set, this will be 0x2FF, meaning the component ID is an unknown plugin type.
<b>scmLink</b>	<b>String</b>	No	A full URL to the Source Control Management (SCM) system or project for the plugin type.
<b>title</b>	<b>String</b>	No	A short name describing the plugin type.
<b>website</b>	<b>String</b>	No	The website of the author.

Related topics

- [Plugin types on page 1607](#)

Fields: Policies

 See the associated [Policies on page 1611](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	policies	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	policies.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

Policy

A Moab Workload Manager policy which can affect scheduling decisions such as resource allocation. A policy contains state, identifying information, a priority, and metadata about the policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

AutoVMMigrationPolicy

The Moab policy used to enabled and configure policy-based VM migration. Using information about data center applications and server load, Moab can aim to keep VMs in the data center optimally distributed across all hypervisors.

This class inherits fields from Policy.

Field Name	Type	P U T	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>genericMetricThresholds</b>	<b>Map&lt;String, Double&gt;</b>	Yes	A map of generic metric pairings where each value must be greater than or equal to 0 such as:  METRIC1 => 5.6 METRIC2 => 0.0 METRIC3 => 102.4
<b>memoryUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the utilization threshold for memory. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>migrationAlgorithmType</b>	<u><a href="#">AutoVMMigrationPolicyType</a></u>	Yes	Configures the VM migration algorithm utilized when the policy is active. Defaults to <u><a href="#">NONE</a></u> . When <u><a href="#">ENABLED</a></u> , this must <b>not</b> be set to NONE.

Field Name	Type	P U T	Description
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>processorUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the load utilization threshold for processors. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

#### [AutoVMMigrationPolicyType](#)

Represents the algorithm used to migrate VMs when the [AutoVMMigrationPolicy](#) is used.

Value	Description
<b>NONE</b>	Used when the Auto VM Migration policy is currently disabled in Moab and before any settings are saved the first time.  For example, if the policy is disabled on the first read of Moab policies, the <a href="#">AutoVMMigrationPolicy.migrationAlgorithmType</a> will be set to NONE. If the policy is enabled and the type is set to OVERCOMMIT, followed by a disabling of the policy, it will then be represented as having a state of DISABLED with a migrationAlgorithmType of OVERCOMMIT.
<b>OVERCOMMIT</b>	Use the "overcommit" algorithm for migration. The goal of this algorithm is to equalize loads across hypervisors as migrations are queued due to overcommit conditions. This places VMs to be migrated on the least-loaded hypervisor available.
<b>CONSOLIDATION</b>	Use the "consolidation" algorithm for migration. The goal of this algorithm is to load hypervisors as close to thresholds as possible, without exceeding them. This policy places VMs to be migrated on the most loaded hypervisor possible, within these constraints. A second loop of this policy will select lightly-loaded hypervisors to be evacuated completely.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

HVAllocationOvercommitPolicy

The Hypervisor Allocation Overcommit policy controls how many virtual machines can be placed on a hypervisor. By enabling this policy, you are allowing Moab to allocate more resources to a set of virtual machines than a hypervisor may actually have. This is possible due to virtualization. In other words, this policy allows you to set the high-water mark for virtual machine allocation for hypervisors. At least one of these limits must be greater than 1.0, or the policy will not be able to set to a state of [PolicyState.ENABLED](#).

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>memoryAllocationLimit</b>	<b>Double</b>	Yes	Setting this to 1 effectively disables the allocation overcommit based on memory. If this and <a href="#">processorAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a> .
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>processorAllocationLimit</b>	<b>Double</b>	Yes	<p>The Allocation Limit defines the upper bound or maximum amount of VCPUs that can be created on any given hypervisor (HV). For example, if you have a hypervisor with 12 processors or cores (Moab sees them as 12 processors), and have an Allocation Limit of 2.0 for procs, then Moab will not allow, under any condition, more than 24 VCPU's to be allocated on this hypervisor. Remember: a VM can have one or more VCPU's. So, in this example, the HV could only support 8 VM's if they all had 3 VPCU's each. It could support 4 VM's if they had 6 VPCU's each, and so forth</p> <p>From <a href="http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html">http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html</a></p> <p>Setting this to 1 effectively disables the allocation overcommit based on processors. If this and <a href="#">memoryAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a>.</p>
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

NodeAllocationPolicy

Node allocation is the process of selecting the best resources to allocate to a job from a list of available resources. Moab contains a number of allocation algorithms that address this in the NodeAllocationPolicy.

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>customPriorityFunction</b>	<b>String</b>	Yes	Defines the priority function when the <a href="#">CustomPriority</a> algorithm is used.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>nodeAllocationAlgorithm</b>	<a href="#">NodeAllocationAlgorithm</a>	Yes	Configures the node allocation algorithm utilized when the policy is active. Defaults to <b>NONE</b> . When <a href="#">ENABLED</a> , this must <b>not</b> be set to NONE.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	Set<String>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	Set<String>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

[NodeAllocationAlgorithm](#)

Represents the algorithm used to allocate Nodes when the [NodeAllocationPolicy](#) is used.

Value	Description
<b>NONE</b>	
<b>InReportedOrder</b>	
<b>InReverseReportedOrder</b>	
<b>CustomPriority</b>	
<b>ProcessorLoad</b>	
<b>MinimumConfiguredResources</b>	
<b>Contiguous</b>	
<b>ProcessorSpeedBalance</b>	
<b>NodeSpeed</b>	

[PolicyState](#)

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

MigrationExclusionListPolicy

Specify which virtual machines and hypervisors to exclude from automatic migration operations.

This class inherits fields from Policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>hvExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of hypervisor IDs on the exclusion list.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<u><a href="#">PolicyState</a></u>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <u><a href="#">PolicyState.DISABLED</a></u> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.

Field Name	Type	PUT	Description
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.
<b>vmExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of VM IDs on the exclusion list.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

FairsharePolicy

Fairshare allows historical resource utilization information to be incorporated into job feasibility and priority decisions. This feature allows site administrators to set system utilization targets for users, groups, accounts, classes, and QoS levels. Administrators can also specify the time frame over which resource utilization is evaluated in determining whether the goal is being reached. Parameters allow sites to specify the utilization metric, how historical information is aggregated, and the effect of fairshare state on scheduling behavior. You can specify fairshare targets for any credentials (such as user, group, and class) that administrators want such information to affect. <http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php>

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.

Field Name	Type	PUT	Description
<b>decayFactor</b>	<b>Double</b>	Yes	Specifies decay rate applied to past fairshare interval when computing effective fairshare usage. Values may be in the range of 0.01 to 1.0. A smaller value causes more rapid decay causing aged usage to contribute less to the overall effective fairshare usage. A value of 1.0 indicates that no decay will occur and all fairshare intervals will be weighted equally when determining effective fairshare usage.
<b>depth</b>	<b>Integer</b>	Yes	Number of fairshare windows factored into current fairshare utilization. Note: The number of available fairshare windows is bounded by the MAX_FSDEPTH value (32 in Moab).
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>intervalSeconds</b>	<b>Long</b>	Yes	Specifies the length of each fairshare window in seconds.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#"><u>PolicyState</u></a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#"><u>PolicyState.DISABLED</u></a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

Field Name	Type	PUT	Description
<b>usageMetric</b>	<a href="#">FairshareUsageMetric</a>	Yes	As Moab runs, it records how available resources are used. Each iteration it updates fairshare resource utilization statistics. Resource utilization is tracked in accordance with the usage metric allowing various aspects of resource consumption information to be measured. The usage metric allows selection of both the types of resources to be tracked as well as the method of tracking. It provides the option of tracking usage by dedicated or consumed resources, where dedicated usage tracks what the scheduler assigns to the job and consumed usage tracks what the job actually uses.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

FairshareUsageMetric

Specifies the unit of tracking [FairsharePolicy](#) usage.

<http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php#fspolicy>

Value	Description
<b>NONE</b>	
<b>DEDICATED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds dedicated to each job relative to other processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)
<b>DEDICATED_PROCESSOR_SECONDS_AVAILABLE</b>	Usage tracked by processor seconds dedicated to each job relative to all available processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)

Value	Description
<b>DEDICATED_PROCESSOR_EQUIVALENT_SECONDS_DELIVERED</b>	Usage tracked by processor-equivalent seconds dedicated to each job relative to other processor-equivalent seconds dedicated to other jobs on the system. (Useful in dedicated and shared nodes environments).
<b>UTILIZED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds used by each job relative to other processor seconds used by other jobs on the system. (Useful in shared node/SMP environments.)

## API version 2

Policy

A Moab Workload Manager policy which can affect scheduling decisions such as resource allocation. A policy contains state, identifying information, a priority, and metadata about the policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

AutoVMMigrationPolicy

The Moab policy used to enabled and configure policy-based VM migration. Using information about data center applications and server load, Moab can aim to keep VMs in the data center optimally distributed across all hypervisors.

This class inherits fields from Policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>genericMetricThresholds</b>	<b>Map&lt;String, Double&gt;</b>	Yes	A map of generic metric pairings where each value must be greater than or equal to 0 such as:  METRIC1 => 5.6 METRIC2 => 0.0 METRIC3 => 102.4
<b>memoryUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the utilization threshold for memory. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>migrationAlgorithmType</b>	<u><a href="#">AutoVMMigrationPolicyType</a></u>	Yes	Configures the VM migration algorithm utilized when the policy is active. Defaults to <u><a href="#">NONE</a></u> . When <u><a href="#">ENABLED</a></u> , this must <b>not</b> be set to NONE.

Field Name	Type	P U T	Description
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>processorUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the load utilization threshold for processors. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

#### [AutoVMMigrationPolicyType](#)

Represents the algorithm used to migrate VMs when the [AutoVMMigrationPolicy](#) is used.

Value	Description
<b>NONE</b>	Used when the Auto VM Migration policy is currently disabled in Moab and before any settings are saved the first time.  For example, if the policy is disabled on the first read of Moab policies, the <a href="#">AutoVMMigrationPolicy.migrationAlgorithmType</a> will be set to NONE. If the policy is enabled and the type is set to OVERCOMMIT, followed by a disabling of the policy, it will then be represented as having a state of DISABLED with a migrationAlgorithmType of OVERCOMMIT.
<b>OVERCOMMIT</b>	Use the "overcommit" algorithm for migration. The goal of this algorithm is to equalize loads across hypervisors as migrations are queued due to overcommit conditions. This places VMs to be migrated on the least-loaded hypervisor available.
<b>CONSOLIDATION</b>	Use the "consolidation" algorithm for migration. The goal of this algorithm is to load hypervisors as close to thresholds as possible, without exceeding them. This policy places VMs to be migrated on the most loaded hypervisor possible, within these constraints. A second loop of this policy will select lightly-loaded hypervisors to be evacuated completely.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

HVAllocationOvercommitPolicy

The Hypervisor Allocation Overcommit policy controls how many virtual machines can be placed on a hypervisor. By enabling this policy, you are allowing Moab to allocate more resources to a set of virtual machines than a hypervisor may actually have. This is possible due to virtualization. In other words, this policy allows you to set the high-water mark for virtual machine allocation for hypervisors. At least one of these limits must be greater than 1.0, or the policy will not be able to set to a state of [PolicyState.ENABLED](#).

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>memoryAllocationLimit</b>	<b>Double</b>	Yes	Setting this to 1 effectively disables the allocation overcommit based on memory. If this and <a href="#">processorAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a> .
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>processorAllocationLimit</b>	<b>Double</b>	Yes	The Allocation Limit defines the upper bound or maximum amount of VCPUs that can be created on any given hypervisor (HV). For example, if you have a hypervisor with 12 processors or cores (Moab sees them as 12 processors), and have an Allocation Limit of 2.0 for procs, then Moab will not allow, under any condition, more than 24 VCPU's to be allocated on this hypervisor. Remember: a VM can have one or more VCPU's. So, in this example, the HV could only support 8 VM's if they all had 3 VPCU's each. It could support 4 VM's if they had 6 VPCU's each, and so forth  From <a href="http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html">http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html</a> Setting this to 1 effectively disables the allocation overcommit based on processors. If this and <a href="#">memoryAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a> .
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

NodeAllocationPolicy

Node allocation is the process of selecting the best resources to allocate to a job from a list of available resources. Moab contains a number of allocation algorithms that address this in the NodeAllocationPolicy.

This class inherits fields from Policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>customPriorityFunction</b>	<b>String</b>	Yes	Defines the priority function when the <u>CustomPriority</u> algorithm is used.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>nodeAllocationAlgorithm</b>	<u><a href="#">NodeAllocationAlgorithm</a></u>	Yes	Configures the node allocation algorithm utilized when the policy is active. Defaults to <b>NONE</b> . When <u><a href="#">ENABLED</a></u> , this must <b>not</b> be set to <b>NONE</b> .
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

[NodeAllocationAlgorithm](#)

Represents the algorithm used to allocate Nodes when the [NodeAllocationPolicy](#) is used.

Value	Description
<b>NONE</b>	
<b>InReportedOrder</b>	
<b>InReverseReportedOrder</b>	
<b>CustomPriority</b>	
<b>ProcessorLoad</b>	
<b>MinimumConfiguredResources</b>	
<b>Contiguous</b>	
<b>ProcessorSpeedBalance</b>	
<b>NodeSpeed</b>	

[PolicyState](#)

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

### MigrationExclusionListPolicy

Specify which virtual machines and hypervisors to exclude from automatic migration operations.

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>hvExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of hypervisor IDs on the exclusion list.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.

Field Name	Type	PUT	Description
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.
<b>vmExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of VM IDs on the exclusion list.

### PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

### FairsharePolicy

Fairshare allows historical resource utilization information to be incorporated into job feasibility and priority decisions. This feature allows site administrators to set system utilization targets for users, groups, accounts, classes, and QoS levels. Administrators can also specify the time frame over which resource utilization is evaluated in determining whether the goal is being reached. Parameters allow sites to specify the utilization metric, how historical information is aggregated, and the effect of fairshare state on scheduling behavior. You can specify fairshare targets for any credentials (such as user, group, and class) that administrators want such information to affect. <http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php>

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.

Field Name	Type	PUT	Description
<b>decayFactor</b>	<b>Double</b>	Yes	Specifies decay rate applied to past fairshare interval when computing effective fairshare usage. Values may be in the range of 0.01 to 1.0. A smaller value causes more rapid decay causing aged usage to contribute less to the overall effective fairshare usage. A value of 1.0 indicates that no decay will occur and all fairshare intervals will be weighted equally when determining effective fairshare usage.
<b>depth</b>	<b>Integer</b>	Yes	Number of fairshare windows factored into current fairshare utilization. Note: The number of available fairshare windows is bounded by the MAX_FSDEPTH value (32 in Moab).
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>intervalSeconds</b>	<b>Long</b>	Yes	Specifies the length of each fairshare window in seconds.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#"><u>PolicyState</u></a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#"><u>PolicyState.DISABLED</u></a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

Field Name	Type	PUT	Description
<b>usageMetric</b>	<a href="#">FairshareUsageMetric</a>	Yes	As Moab runs, it records how available resources are used. Each iteration it updates fairshare resource utilization statistics. Resource utilization is tracked in accordance with the usage metric allowing various aspects of resource consumption information to be measured. The usage metric allows selection of both the types of resources to be tracked as well as the method of tracking. It provides the option of tracking usage by dedicated or consumed resources, where dedicated usage tracks what the scheduler assigns to the job and consumed usage tracks what the job actually uses.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

FairshareUsageMetric

Specifies the unit of tracking [FairsharePolicy](#) usage.

<http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php#fspolicy>

Value	Description
<b>NONE</b>	
<b>DEDICATED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds dedicated to each job relative to other processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)
<b>DEDICATED_PROCESSOR_SECONDS_AVAILABLE</b>	Usage tracked by processor seconds dedicated to each job relative to all available processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)

Value	Description
<b>DEDICATED_PROCESSOR_EQUIVALENT_SECONDS_DELIVERED</b>	Usage tracked by processor-equivalent seconds dedicated to each job relative to other processor-equivalent seconds dedicated to other jobs on the system. (Useful in dedicated and shared nodes environments).
<b>UTILIZED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds used by each job relative to other processor seconds used by other jobs on the system. (Useful in shared node/SMP environments.)

## API version 1

Policy

A Moab Workload Manager policy which can affect scheduling decisions such as resource allocation. A policy contains state, identifying information, a priority, and metadata about the policy.

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

AutoVMMigrationPolicy

The Moab policy used to enabled and configure policy-based VM migration. Using information about data center applications and server load, Moab can aim to keep VMs in the data center optimally distributed across all hypervisors.

This class inherits fields from Policy.

Field Name	Type	P U T	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>genericMetricThresholds</b>	<b>Map&lt;String, Double&gt;</b>	Yes	A map of generic metric pairings where each value must be greater than or equal to 0 such as:  METRIC1 => 5.6 METRIC2 => 0.0 METRIC3 => 102.4
<b>memoryUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the utilization threshold for memory. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>migrationAlgorithmType</b>	<u><a href="#">AutoVMMigrationPolicyType</a></u>	Yes	Configures the VM migration algorithm utilized when the policy is active. Defaults to <u><a href="#">NONE</a></u> . When <u><a href="#">ENABLED</a></u> , this must <b>not</b> be set to NONE.

Field Name	Type	P U T	Description
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>processorUtilizationThreshold</b>	<b>Double</b>	Yes	Defines the load utilization threshold for processors. This must be greater than 0 and less than or equal to 1. A value of 1 effectively disables the threshold.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

#### [AutoVMMigrationPolicyType](#)

Represents the algorithm used to migrate VMs when the [AutoVMMigrationPolicy](#) is used.

Value	Description
<b>NONE</b>	Used when the Auto VM Migration policy is currently disabled in Moab and before any settings are saved the first time.  For example, if the policy is disabled on the first read of Moab policies, the <a href="#">AutoVMMigrationPolicy.migrationAlgorithmType</a> will be set to NONE. If the policy is enabled and the type is set to OVERCOMMIT, followed by a disabling of the policy, it will then be represented as having a state of DISABLED with a migrationAlgorithmType of OVERCOMMIT.
<b>OVERCOMMIT</b>	Use the "overcommit" algorithm for migration. The goal of this algorithm is to equalize loads across hypervisors as migrations are queued due to overcommit conditions. This places VMs to be migrated on the least-loaded hypervisor available.
<b>CONSOLIDATION</b>	Use the "consolidation" algorithm for migration. The goal of this algorithm is to load hypervisors as close to thresholds as possible, without exceeding them. This policy places VMs to be migrated on the most loaded hypervisor possible, within these constraints. A second loop of this policy will select lightly-loaded hypervisors to be evacuated completely.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

HVAllocationOvercommitPolicy

The Hypervisor Allocation Overcommit policy controls how many virtual machines can be placed on a hypervisor. By enabling this policy, you are allowing Moab to allocate more resources to a set of virtual machines than a hypervisor may actually have. This is possible due to virtualization. In other words, this policy allows you to set the high-water mark for virtual machine allocation for hypervisors. At least one of these limits must be greater than 1.0, or the policy will not be able to set to a state of [PolicyState.ENABLED](#).

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>memoryAllocationLimit</b>	<b>Double</b>	Yes	Setting this to 1 effectively disables the allocation overcommit based on memory. If this and <a href="#">processorAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a> .
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>processorAllocationLimit</b>	<b>Double</b>	Yes	<p>The Allocation Limit defines the upper bound or maximum amount of VCPUs that can be created on any given hypervisor (HV). For example, if you have a hypervisor with 12 processors or cores (Moab sees them as 12 processors), and have an Allocation Limit of 2.0 for procs, then Moab will not allow, under any condition, more than 24 VCPU's to be allocated on this hypervisor. Remember: a VM can have one or more VCPU's. So, in this example, the HV could only support 8 VM's if they all had 3 VPCU's each. It could support 4 VM's if they had 6 VPCU's each, and so forth</p> <p>From <a href="http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html">http://www.adaptivecomputing.com/resources/docs/mwm/7-1-1/Content/topics/vm/allocation_limits_and_utilization_threshold.html</a></p> <p>Setting this to 1 effectively disables the allocation overcommit based on processors. If this and <a href="#">memoryAllocationLimit</a> are both set to 1.0 (the default), the policy state cannot be set to <a href="#">PolicyState.ENABLED</a>.</p>
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

NodeAllocationPolicy

Node allocation is the process of selecting the best resources to allocate to a job from a list of available resources. Moab contains a number of allocation algorithms that address this in the NodeAllocationPolicy.

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>customPriorityFunction</b>	<b>String</b>	Yes	Defines the priority function when the <a href="#">CustomPriority</a> algorithm is used.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>nodeAllocationAlgorithm</b>	<a href="#">NodeAllocationAlgorithm</a>	Yes	Configures the node allocation algorithm utilized when the policy is active. Defaults to <b>NONE</b> . When <a href="#">ENABLED</a> , this must <b>not</b> be set to NONE.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.

Field Name	Type	PUT	Description
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	Set<String>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	Set<String>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

NodeAllocationAlgorithm

Represents the algorithm used to allocate Nodes when the [NodeAllocationPolicy](#) is used.

Value	Description
<b>NONE</b>	
<b>InReportedOrder</b>	
<b>InReverseReportedOrder</b>	
<b>CustomPriority</b>	
<b>ProcessorLoad</b>	
<b>MinimumConfiguredResources</b>	
<b>Contiguous</b>	
<b>ProcessorSpeedBalance</b>	
<b>NodeSpeed</b>	

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

### MigrationExclusionListPolicy

Specify which virtual machines and hypervisors to exclude from automatic migration operations.

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>hvExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of hypervisor IDs on the exclusion list.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#">PolicyState</a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#">PolicyState.DISABLED</a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.

Field Name	Type	PUT	Description
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.
<b>vmExclusionList</b>	<b>List&lt;String&gt;</b>	Yes	The list of VM IDs on the exclusion list.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

FairsharePolicy

Fairshare allows historical resource utilization information to be incorporated into job feasibility and priority decisions. This feature allows site administrators to set system utilization targets for users, groups, accounts, classes, and QoS levels. Administrators can also specify the time frame over which resource utilization is evaluated in determining whether the goal is being reached. Parameters allow sites to specify the utilization metric, how historical information is aggregated, and the effect of fairshare state on scheduling behavior. You can specify fairshare targets for any credentials (such as user, group, and class) that administrators want such information to affect. <http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php>

This class inherits fields from [Policy](#).

Field Name	Type	PUT	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the policy. Must contain only lowercase letters and dashes, such as 'auto-vm-migration'.
<b>conflicted</b>	<b>Boolean</b>	No	Signifies whether any other policies are currently activated that potentially conflict with this policy. If true, it signifies a <b>potential</b> conflict.

Field Name	Type	PUT	Description
<b>decayFactor</b>	<b>Double</b>	Yes	Specifies decay rate applied to past fairshare interval when computing effective fairshare usage. Values may be in the range of 0.01 to 1.0. A smaller value causes more rapid decay causing aged usage to contribute less to the overall effective fairshare usage. A value of 1.0 indicates that no decay will occur and all fairshare intervals will be weighted equally when determining effective fairshare usage.
<b>depth</b>	<b>Integer</b>	Yes	Number of fairshare windows factored into current fairshare utilization. Note: The number of available fairshare windows is bounded by the MAX_FSDEPTH value (32 in Moab).
<b>description</b>	<b>String</b>	No	The user friendly description of the policy.
<b>intervalSeconds</b>	<b>Long</b>	Yes	Specifies the length of each fairshare window in seconds.
<b>name</b>	<b>String</b>	No	The user friendly name of the policy.
<b>potentialConflicts</b>	<b>Set&lt;String&gt;</b>	No	A set of policy IDs that may potentially conflict with this policy.
<b>priority</b>	<b>Integer</b>	No	Indicates the absolute priority of the policy with respect to others. It is possible that more than one policy has the same priority. The higher the number, the greater the priority. Minimum is 0.
<b>state</b>	<a href="#"><u>PolicyState</u></a>	Yes	Defines the current state of the policy: enabled or disabled. Defaults to <a href="#"><u>PolicyState.DISABLED</u></a> .
<b>tags</b>	<b>Set&lt;String&gt;</b>	No	A set of strings that can be used to aid in filtering or querying policies.
<b>types</b>	<b>Set&lt;String&gt;</b>	No	A set of categories or types that the policy is included in. This may be used to filter or query on groups of policies.

Field Name	Type	PUT	Description
<b>usageMetric</b>	<a href="#">FairshareUsageMetric</a>	Yes	As Moab runs, it records how available resources are used. Each iteration it updates fairshare resource utilization statistics. Resource utilization is tracked in accordance with the usage metric allowing various aspects of resource consumption information to be measured. The usage metric allows selection of both the types of resources to be tracked as well as the method of tracking. It provides the option of tracking usage by dedicated or consumed resources, where dedicated usage tracks what the scheduler assigns to the job and consumed usage tracks what the job actually uses.

PolicyState

Represents the state of a policy. A policy may only be enabled or disabled.

Value	Description
<b>ENABLED</b>	The policy is enabled or active.
<b>DISABLED</b>	The policy is disabled or inactive.

FairshareUsageMetric

Specifies the unit of tracking [FairsharePolicy](#) usage.

<http://docs.adaptivecomputing.com/mwm/archive/6-0-4/6.3fairshare.php#fspolicy>

Value	Description
<b>NONE</b>	
<b>DEDICATED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds dedicated to each job relative to other processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)
<b>DEDICATED_PROCESSOR_SECONDS_AVAILABLE</b>	Usage tracked by processor seconds dedicated to each job relative to all available processor seconds dedicated to other jobs on the system. (Useful in dedicated node environments.)

Value	Description
<b>DEDICATED_PROCESSOR_EQUIVALENT_SECONDS_DELIVERED</b>	Usage tracked by processor-equivalent seconds dedicated to each job relative to other processor-equivalent seconds dedicated to other jobs on the system. (Useful in dedicated and shared nodes environments).
<b>UTILIZED_PROCESSOR_SECONDS_DELIVERED</b>	Usage tracked by processor seconds used by each job relative to other processor seconds used by other jobs on the system. (Useful in shared node/SMP environments.)

Related topics

- [Policies on page 1611](#)

## Fields: Principals

 See the associated [Principals on page 1627](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	principals	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	principals.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

API version 3

Principal

A principal maps to a set of ldap users and/or ldap groups. MWS roles are attached to the principals to authorize the group to use the specific MWS roles.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this principal.
<b>attachedRoles</b>	<a href="#">Set&lt;Role&gt;</a>	Yes	Yes	The MWS roles this principal is authorized to use.
<b>description</b>	<b>String</b>	Yes	Yes	The principal description.
<b>groups</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The groups associated with this principal. Each group has a name and a type. The valid types of groups are LDAPOU, LDAPGROUP, and SSO. Example group: {"name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev", "type": "LDAPGROUP"}
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this principal. Required during POST.
<b>users</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The users associated with this principal. Each user has a name and type. The only valid type of user is LDAP. Example user: {"name": "jhammon", "type": "LDAP"}

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally

Field Name	Type	POST	PUT	Description
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

API version 2

Principal

A principal maps to a set of ldap users and/or ldap groups. MWS roles are attached to the principals to authorize the group to use the specific MWS roles.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this principal.
<b>attachedRoles</b>	<a href="#">Set&lt;Role&gt;</a>	Yes	Yes	The MWS roles this principal is authorized to use.
<b>description</b>	<b>String</b>	Yes	Yes	The principal description.
<b>groups</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The groups associated with this principal. Each group has a name and a type. The valid types of groups are LDAPOU, LDAPGROUP, and SSO. Example group: {"name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev", "type": "LDAPGROUP"}
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this principal. Required during POST.
<b>users</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The users associated with this principal. Each user has a name and type. The only valid type of user is LDAP. Example user: {"name": "jhammon", "type": "LDAP"}

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally

Field Name	Type	POST	PUT	Description
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

API version 1

Principal

A principal maps to a set of ldap users and/or ldap groups. MWS roles are attached to the principals to authorize the group to use the specific MWS roles.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this principal.
<b>attachedRoles</b>	<a href="#">Set&lt;Role&gt;</a>	Yes	Yes	The MWS roles this principal is authorized to use.
<b>description</b>	<b>String</b>	Yes	Yes	The principal description.
<b>groups</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The groups associated with this principal. Each group has a name and a type. The valid types of groups are LDAPOU, LDAPGROUP, and SSO. Example group: {"name": "CN=Engineering,CN=Users,DC=corp,DC=cloud,DC=dev", "type": "LDAPGROUP"}
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this principal. Required during POST.
<b>users</b>	<b>List&lt;Map&gt;</b>	Yes	Yes	The users associated with this principal. Each user has a name and type. The only valid type of user is LDAP. Example user: {"name": "jhammon", "type": "LDAP"}

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally

Field Name	Type	POST	PUT	Description
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

Related topics

- [Principals on page 1627](#)

## Fields: Report Datapoints

**i** See the associated [Reports on page 1636](#) resource section for more information on how to use this resource and supported operations.

### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	reports/datapoints	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	reports.datapoints.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

### API version 3

#### Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	The ending date that the datapoint covers.
<b>firstSampleDate</b>	<b>Date</b>	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>lastSampleDate</b>	<b>Date</b>	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>startDate</b>	<b>Date</b>	The beginning date that the datapoint covers.

## API version 2

Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	The ending date that the datapoint covers.
<b>firstSampleDate</b>	<b>Date</b>	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>lastSampleDate</b>	<b>Date</b>	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>startDate</b>	<b>Date</b>	The beginning date that the datapoint covers.

## API version 1

Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	Description
<b>id</b>	<b>Long</b>	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	The ending date that the datapoint covers.
<b>firstSampleDate</b>	<b>Date</b>	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>lastSampleDate</b>	<b>Date</b>	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>startDate</b>	<b>Date</b>	The beginning date that the datapoint covers.

## Related topics

- [Reports on page 1636](#)

## Fields: Reports

**i** See the associated [Reports on page 1636](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	reports	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	reports.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

API version 3

Report

A set of time-based values that share similar context. For example, a report may contain data on CPU or power utilization for all nodes in a cluster.

A report is composed of metadata and a collection of [Datapoints](#). [Samples](#) are also associated with reports, but these are consolidated using the [Report.consolidationFunction](#) to create [Datapoints](#).

If the datapoint documents are being truncated in any way or there are warnings about documents being too large, it may be necessary to increase the [Report.reportDocumentSize](#).

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the report. This is automatically assigned and will be ignored if specified duration creation.
<b>consolidationFunction</b>	<b>String</b>	Yes	The consolidation function is the process used to convert a set of samples into a datapoint. Currently the only supported function is "average", which is used if none is specified.
<b>datapointDuration</b>	<b>Long</b>	Yes	Required. How long the datapoints are, in seconds.
<b>datapoints</b>	<a href="#">List&lt;Datapoint&gt;</a>	Yes	This is the set of datapoints that have been consolidated for the report or are desired to be included in the report during creation time. In the latter case, these represent historical data created outside of the reporting framework. Only present when getting a single report.
<b>description</b>	<b>String</b>	Yes	A description of the report.
<b>keepSamples</b>	<b>Boolean</b>	Yes	Controls if samples are retained after consolidation. Defaults to false, which means that after consolidation, samples are discarded.
<b>minimumSampleSize</b>	<b>Integer</b>	Yes	If number of samples is below this number, the datapoint data field is "null". Defaults to 1.
<b>name</b>	<b>String</b>	Yes	Required. A unique name identifying the report. Valid characters are all alphanumeric characters, dashes (-), periods (.), and underscores (_).

Field Name	Type	POST	Description
<b>reportDocumentSize</b>	<b>Long</b>	Yes	<p>The maximum size in bytes of each datapoint document stored for this report. This option is provided to maximize the amount of disk space used for a single report. The default value for this option is 100*1024, or 100 KB. The maximum value of this option is 16*1024*1024 (16777216) or 16 MB, which represents the maximum document size in MongoDB. See also <a href="http://www.mongodb.org/display/DOCS/Documents">http://www.mongodb.org/display/DOCS/Documents</a>.</p> <p>Keep in mind that when creating a new report, MongoDB will initialize all needed space for all possible datapoint documents up front. This can easily fill a disk unless this parameter is modified.</p>
<b>reportSize</b>	<b>Long</b>	Yes	<p>Required. The size of the report in datapoints. After this number of datapoints is reached, the old datapoints will be discarded.</p> <p><b>WARNING:</b> On report creation, a Mongo collection will be initialized that is the maximum size of a single entry (currently 16 MB) multiplied by the report size. Be careful in setting a large report size as this will quickly allocate the entire disk if many reports with large report sizes are created.</p>

### Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	No	The ending date that the datapoint covers.

Field Name	Type	POST	Description
<b>firstSampleDate</b>	<b>Date</b>	No	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp.</a> )
<b>lastSampleDate</b>	<b>Date</b>	No	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp.</a> )
<b>startDate</b>	<b>Date</b>	No	The beginning date that the datapoint covers.

## API version 2

Report

A set of time-based values that share similar context. For example, a report may contain data on CPU or power utilization for all nodes in a cluster.

A report is composed of metadata and a collection of [Datapoints](#). [Samples](#) are also associated with reports, but these are consolidated using the [Report.consolidationFunction](#) to create [Datapoints](#).

If the datapoint documents are being truncated in any way or there are warnings about documents being too large, it may be necessary to increase the [Report.reportDocumentSize](#).

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the report. This is automatically assigned and will be ignored if specified duration creation.
<b>consolidationFunction</b>	<b>String</b>	Yes	The consolidation function is the process used to convert a set of samples into a datapoint. Currently the only supported function is "average", which is used if none is specified.
<b>datapointDuration</b>	<b>Long</b>	Yes	Required. How long the datapoints are, in seconds.
<b>datapoints</b>	<a href="#">List&lt;Datapoint&gt;</a>	Yes	This is the set of datapoints that have been consolidated for the report or are desired to be included in the report during creation time. In the latter case, these represent historical data created outside of the reporting framework. Only present when getting a single report.
<b>description</b>	<b>String</b>	Yes	A description of the report.
<b>keepSamples</b>	<b>Boolean</b>	Yes	Controls if samples are retained after consolidation. Defaults to false, which means that after consolidation, samples are discarded.
<b>minimumSampleSize</b>	<b>Integer</b>	Yes	If number of samples is below this number, the datapoint data field is "null". Defaults to 1.
<b>name</b>	<b>String</b>	Yes	Required. A unique name identifying the report. Valid characters are all alphanumeric characters, dashes (-), periods (.), and underscores (_).

Field Name	Type	POST	Description
<b>reportDocumentSize</b>	<b>Long</b>	Yes	<p>The maximum size in bytes of each datapoint document stored for this report. This option is provided to maximize the amount of disk space used for a single report. The default value for this option is 100*1024, or 100 KB. The maximum value of this option is 16*1024*1024 (16777216) or 16 MB, which represents the maximum document size in MongoDB. See also <a href="http://www.mongodb.org/display/DOCS/Documents">http://www.mongodb.org/display/DOCS/Documents</a>.</p> <p>Keep in mind that when creating a new report, MongoDB will initialize all needed space for all possible datapoint documents up front. This can easily fill a disk unless this parameter is modified.</p>
<b>reportSize</b>	<b>Long</b>	Yes	<p>Required. The size of the report in datapoints. After this number of datapoints is reached, the old datapoints will be discarded.</p> <p>WARNING: On report creation, a Mongo collection will be initialized that is the maximum size of a single entry (currently 16 MB) multiplied by the report size. Be careful in setting a large report size as this will quickly allocate the entire disk if many reports with large report sizes are created.</p>

Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	No	The ending date that the datapoint covers.

Field Name	Type	POST	Description
<b>firstSampleDate</b>	<b>Date</b>	No	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>lastSampleDate</b>	<b>Date</b>	No	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp</a> .)
<b>startDate</b>	<b>Date</b>	No	The beginning date that the datapoint covers.

API version 1

Report

A set of time-based values that share similar context. For example, a report may contain data on CPU or power utilization for all nodes in a cluster.

A report is composed of metadata and a collection of [Datapoints](#). [Samples](#) are also associated with reports, but these are consolidated using the [Report.consolidationFunction](#) to create [Datapoints](#).

If the datapoint documents are being truncated in any way or there are warnings about documents being too large, it may be necessary to increase the [Report.reportDocumentSize](#).

Field Name	Type	POST	Description
<b>id</b>	<b>String</b>	No	The unique identifier for the report. This is automatically assigned and will be ignored if specified duration creation.
<b>consolidationFunction</b>	<b>String</b>	Yes	The consolidation function is the process used to convert a set of samples into a datapoint. Currently the only supported function is "average", which is used if none is specified.
<b>datapointDuration</b>	<b>Long</b>	Yes	Required. How long the datapoints are, in seconds.
<b>datapoints</b>	<a href="#">List&lt;Datapoint&gt;</a>	Yes	This is the set of datapoints that have been consolidated for the report or are desired to be included in the report during creation time. In the latter case, these represent historical data created outside of the reporting framework. Only present when getting a single report.
<b>description</b>	<b>String</b>	Yes	A description of the report.
<b>keepSamples</b>	<b>Boolean</b>	Yes	Controls if samples are retained after consolidation. Defaults to false, which means that after consolidation, samples are discarded.
<b>minimumSampleSize</b>	<b>Integer</b>	Yes	If number of samples is below this number, the datapoint data field is "null". Defaults to 1.
<b>name</b>	<b>String</b>	Yes	Required. A unique name identifying the report. Valid characters are all alphanumeric characters, dashes (-), periods (.), and underscores (_).

Field Name	Type	POST	Description
<b>reportDocumentSize</b>	<b>Long</b>	Yes	<p>The maximum size in bytes of each datapoint document stored for this report. This option is provided to maximize the amount of disk space used for a single report. The default value for this option is 100*1024, or 100 KB. The maximum value of this option is 16*1024*1024 (16777216) or 16 MB, which represents the maximum document size in MongoDB. See also <a href="http://www.mongodb.org/display/DOCS/Documents">http://www.mongodb.org/display/DOCS/Documents</a>.</p> <p>Keep in mind that when creating a new report, MongoDB will initialize all needed space for all possible datapoint documents up front. This can easily fill a disk unless this parameter is modified.</p>
<b>reportSize</b>	<b>Long</b>	Yes	<p>Required. The size of the report in datapoints. After this number of datapoints is reached, the old datapoints will be discarded.</p> <p><b>WARNING:</b> On report creation, a Mongo collection will be initialized that is the maximum size of a single entry (currently 16 MB) multiplied by the report size. Be careful in setting a large report size as this will quickly allocate the entire disk if many reports with large report sizes are created.</p>

### Datapoint

A metric that measures system state over a specified period of time. For example, a datapoint may contain data on CPU utilization by specific users. A datapoint is generated by the consolidation of zero or more [Samples](#). It could be said that a datapoint represents a smoothing of samples.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	The actual consolidated sample data. This property may be 'null' if the <a href="#">Report.minimumSampleSize</a> was not met when consolidating the datapoint.
<b>endDate</b>	<b>Date</b>	No	The ending date that the datapoint covers.

Field Name	Type	POST	Description
<b>firstSampleDate</b>	<b>Date</b>	No	The date of the first sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp.</a> )
<b>lastSampleDate</b>	<b>Date</b>	No	The date of the last sample consolidated in this datapoint. (See also: <a href="#">Sample.timestamp.</a> )
<b>startDate</b>	<b>Date</b>	No	The beginning date that the datapoint covers.

### Related topics

- [Reports on page 1636](#)

## Fields: Reservations

**i** See the associated [Reservations on page 1646](#) resource section for more information on how to use this resource and supported operations.

### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	reservations	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	reservations.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Yes	No	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Yes	No	Accountable Group.
<b>accountingQOS</b>	<b>String</b>	Yes	No	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Yes	No	Accountable User.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	Yes	No	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	No	No	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	No	No	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	No	No	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Yes	No	Reservation's comments or description.

Field Name	Type	POST	PUT	Description
<b>creationDate</b>	<b>Date</b>	No	No	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	Yes	No	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	Yes	No	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	No	No	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	Yes	No	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	No	No	Global reservation ID.

Field Name	Type	POST	PUT	Description
<b>hostListExpression</b>	<b>String</b>	Yes	No	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	Yes	No	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	No	No	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	No	No	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	Yes	No	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	No	No	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	No	No	Messages for the reservation.

Field Name	Type	POST	PUT	Description
<b>owner</b>	<a href="#">EmbeddedCredential</a>	Yes	No	The owner of the reservation
<b>partitionId</b>	String	Yes	No	The ID of the partition this reservation is for.
<b>profile</b>	String	Yes	No	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	Yes	No	The reservation's requirements.
<b>reservationGroup</b>	String	Yes	No	The reservation group to which the reservation belongs.
<b>resources</b>	Map<String, Integer>	Yes	No	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	Date	Yes	No	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#">ReservationStatistics</a>	No	No	The reservation's statistical information.
<b>subType</b>	String	Yes	No	The reservation sub-type.

Field Name	Type	POST	PUT	Description
<b>taskCount</b>	<b>Integer</b>	No	No	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#">Trigger</a>	Yes	No	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	No	No	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	No	No	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The set of variables for this reservation.

[AclRule](#)

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	POST	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	No	Yes	<p>Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.</p> <p>Defaults to <a href="#">AclAffinity.POSITIVE</a>.</p>
<b>comparator</b>	<a href="#">ComparisonOperator</a>	No	Yes	<p>The type of comparison to make against the ACL object.</p> <p>Defaults to <a href="#">ComparisonOperator.EQUAL</a>.</p>
<b>type</b>	<a href="#">AclType</a>	No	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	No	Yes	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.

Value	Description
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported

Value	Description
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

DomainProxyVersion1

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The id of the object.

ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOVERLAP</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.

Value	Description
<b>NOVMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER==ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRE</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.

Value	Description
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.

Value	Description
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.
<b>TRIGHASFIRED</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

#### MessageVersion1

Field Name	Type	POST	PUT	Description
<b>author</b>	<b>String</b>	No	No	The author of the message.
<b>creationTime</b>	<b>Date</b>	No	No	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	No	No	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	No	No	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	No	Yes	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	No	No	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	No	No	An optional priority that can be attached to the comment.

#### EmbeddedCredential

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>type</b>	<a href="#">CredentialType</a>	No	No	

CredentialType

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	POST	PUT	Description
<b>architecture</b>	<b>String</b>	Yes	No	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	No	No	Required feature mode.
<b>memory</b>	<b>Integer</b>	Yes	No	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	No	No	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	No	No	The list of node IDs required for this reservation.

Field Name	Type	POST	PUT	Description
<b>os</b>	<b>String</b>	Yes	No	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Yes	No	Required task count.

ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	POST	PUT	Description
<b>caps</b>	<b>Double</b>	No	No	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	No	No	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	No	No	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	No	No	The total idle processor-seconds over the life of the reservation.

Trigger

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	No	No	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).

Field Name	Type	POST	PUT	Description
<b>actionType</b>	<a href="#">TriggerActionType</a>	No	No	
<b>blockTime</b>	Date	No	No	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	String	No	No	
<b>eventType</b>	<a href="#">TriggerEventType</a>	No	No	
<b>expireTime</b>	Date	No	No	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	Date	No	No	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	No	No	
<b>interval</b>	Boolean	No	No	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	Integer	No	No	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	Boolean	No	No	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	String	No	No	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	String	No	No	The ID of the object which this is attached to.

Field Name	Type	POST	PUT	Description
<b>objectType</b>	<b>String</b>	No	No	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	No	No	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	No	No	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	No	No	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	No	No	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	No	No	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.

Field Name	Type	POST	PUT	Description
<b>threshold</b>	<b>String</b>	No	No	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	No	No	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	No	No	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.

Value	Description
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.

Value	Description
<b>SOFT_KILL</b>	<p>By default, a <code>SIGKILL</code> (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a <code>SIGTERM</code> (kill -15) signal to kill the script. The <code>SIGTERM</code> signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the <code>SIGKILL</code> signal).</p> <p>NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the <code>SIGTERM</code> signal, and never the <code>SIGKILL</code> signal.</p>

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

## API version 2

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Yes	No	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Yes	No	Accountable Group.
<b>accountingQOS</b>	<b>String</b>	Yes	No	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Yes	No	Accountable User.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	Yes	No	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	No	No	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	No	No	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	No	No	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Yes	No	Reservation's comments or description.

Field Name	Type	POST	PUT	Description
<b>creationDate</b>	<b>Date</b>	No	No	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	Yes	No	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	Yes	No	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	No	No	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	Yes	No	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	No	No	Global reservation ID.

Field Name	Type	POST	PUT	Description
<b>hostListExpression</b>	<b>String</b>	Yes	No	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	Yes	No	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	No	No	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	No	No	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	Yes	No	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	No	No	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	No	No	Messages for the reservation.

Field Name	Type	POST	PUT	Description
<b>owner</b>	<a href="#">EmbeddedCredential</a>	Yes	No	The owner of the reservation
<b>partitionId</b>	String	Yes	No	The ID of the partition this reservation is for.
<b>profile</b>	String	Yes	No	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	Yes	No	The reservation's requirements.
<b>reservationGroup</b>	String	Yes	No	The reservation group to which the reservation belongs.
<b>resources</b>	Map<String, Integer>	Yes	No	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	Date	Yes	No	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#">ReservationStatistics</a>	No	No	The reservation's statistical information.
<b>subType</b>	String	Yes	No	The reservation sub-type.

Field Name	Type	POST	PUT	Description
<b>taskCount</b>	<b>Integer</b>	No	No	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#">Trigger</a>	Yes	No	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	No	No	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	No	No	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The set of variables for this reservation.

[AclRule](#)

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	POST	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	No	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#">AclAffinity.POSITIVE</a> .
<b>comparator</b>	<a href="#">ComparisonOperator</a>	No	Yes	The type of comparison to make against the ACL object.  Defaults to <a href="#">ComparisonOperator.EQUAL</a> .
<b>type</b>	<a href="#">AclType</a>	No	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	No	Yes	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.

Value	Description
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported

Value	Description
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

DomainProxyVersion1

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The id of the object.

ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOVERLAP</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.

Value	Description
<b>NOVMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER==ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRE</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.

Value	Description
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.

Value	Description
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.
<b>TRIGHASFIRED</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

#### MessageVersion1

Field Name	Type	POST	PUT	Description
<b>author</b>	<b>String</b>	No	No	The author of the message.
<b>creationTime</b>	<b>Date</b>	No	No	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	No	No	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	No	No	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	No	Yes	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	No	No	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	No	No	An optional priority that can be attached to the comment.

#### EmbeddedCredential

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>type</b>	<a href="#">CredentialType</a>	No	No	

CredentialType

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	POST	PUT	Description
<b>architecture</b>	<b>String</b>	Yes	No	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	No	No	Required feature mode.
<b>memory</b>	<b>Integer</b>	Yes	No	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	No	No	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	No	No	The list of node IDs required for this reservation.

Field Name	Type	POST	PUT	Description
<b>os</b>	<b>String</b>	Yes	No	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Yes	No	Required task count.

### ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	POST	PUT	Description
<b>caps</b>	<b>Double</b>	No	No	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	No	No	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	No	No	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	No	No	The total idle processor-seconds over the life of the reservation.

### Trigger

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	No	No	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).

Field Name	Type	POST	PUT	Description
<b>actionType</b>	<a href="#">TriggerActionType</a>	No	No	
<b>blockTime</b>	Date	No	No	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	String	No	No	
<b>eventType</b>	<a href="#">TriggerEventType</a>	No	No	
<b>expireTime</b>	Date	No	No	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	Date	No	No	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	No	No	
<b>interval</b>	Boolean	No	No	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	Integer	No	No	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	Boolean	No	No	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	String	No	No	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	String	No	No	The ID of the object which this is attached to.

Field Name	Type	POST	PUT	Description
<b>objectType</b>	<b>String</b>	No	No	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	No	No	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	No	No	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	No	No	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	No	No	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	No	No	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.

Field Name	Type	POST	PUT	Description
<b>threshold</b>	<b>String</b>	No	No	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	No	No	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	No	No	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.

Value	Description
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.

Value	Description
<b>SOFT_KILL</b>	<p>By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).</p> <p>NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.</p>

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

## API version 1

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Yes	No	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Yes	No	Accountable Group.
<b>accountingQOS</b>	<b>String</b>	Yes	No	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Yes	No	Accountable User.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	Yes	No	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	No	No	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#">Set&lt;DomainProxyVersion1&gt;</a>	No	No	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	No	No	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	No	No	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Yes	No	Reservation's comments or description.

Field Name	Type	POST	PUT	Description
<b>creationDate</b>	<b>Date</b>	No	No	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	Yes	No	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	Yes	No	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	No	No	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	Yes	No	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	No	No	Global reservation ID.

Field Name	Type	POST	PUT	Description
<b>hostListExpression</b>	<b>String</b>	Yes	No	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	Yes	No	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	No	No	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	No	No	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	Yes	No	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	No	No	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	No	No	Messages for the reservation.

Field Name	Type	POST	PUT	Description
<b>owner</b>	<a href="#">EmbeddedCredential</a>	Yes	No	The owner of the reservation
<b>partitionId</b>	String	Yes	No	The ID of the partition this reservation is for.
<b>profile</b>	String	Yes	No	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	Yes	No	The reservation's requirements.
<b>reservationGroup</b>	String	Yes	No	The reservation group to which the reservation belongs.
<b>resources</b>	Map<String, Integer>	Yes	No	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	Date	Yes	No	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#">ReservationStatistics</a>	No	No	The reservation's statistical information.
<b>subType</b>	String	Yes	No	The reservation sub-type.

Field Name	Type	POST	PUT	Description
<b>taskCount</b>	<b>Integer</b>	No	No	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#">Trigger</a>	Yes	No	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	No	No	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	No	No	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	Yes	Yes	The set of variables for this reservation.

[AclRule](#)

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	POST	PUT	Description
<b>affinity</b>	<a href="#">AclAffinity</a>	No	Yes	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#">AclAffinity.POSITIVE</a> .
<b>comparator</b>	<a href="#">ComparisonOperator</a>	No	Yes	The type of comparison to make against the ACL object.  Defaults to <a href="#">ComparisonOperator.EQUAL</a> .
<b>type</b>	<a href="#">AclType</a>	No	Yes	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	No	Yes	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.

Value	Description
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcIType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported

Value	Description
<b>RACK</b>	Not supported
<b>SCHED</b>	Not supported
<b>SYTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

DomainProxyVersion1

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The id of the object.

ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOVERLAP</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.

Value	Description
<b>NOVMMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER==ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRE</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.

Value	Description
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.

Value	Description
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.
<b>TRIGHASFIRED</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

#### MessageVersion1

Field Name	Type	POST	PUT	Description
<b>author</b>	<b>String</b>	No	No	The author of the message.
<b>creationTime</b>	<b>Date</b>	No	No	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	No	No	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	No	No	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	No	Yes	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	No	No	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	No	No	An optional priority that can be attached to the comment.

#### EmbeddedCredential

Field Name	Type	POST	PUT	Description
<b>name</b>	<b>String</b>	No	No	
<b>type</b>	<a href="#">CredentialType</a>	No	No	

CredentialType

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	POST	PUT	Description
<b>architecture</b>	<b>String</b>	Yes	No	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	Yes	No	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	No	No	Required feature mode.
<b>memory</b>	<b>Integer</b>	Yes	No	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	No	No	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	No	No	The list of node IDs required for this reservation.

Field Name	Type	POST	PUT	Description
<b>os</b>	<b>String</b>	Yes	No	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Yes	No	Required task count.

### ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	POST	PUT	Description
<b>caps</b>	<b>Double</b>	No	No	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	No	No	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	No	No	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	No	No	The total idle processor-seconds over the life of the reservation.

### Trigger

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	No	No	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).

Field Name	Type	POST	PUT	Description
<b>actionType</b>	<a href="#">TriggerActionType</a>	No	No	
<b>blockTime</b>	Date	No	No	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	String	No	No	
<b>eventType</b>	<a href="#">TriggerEventType</a>	No	No	
<b>expireTime</b>	Date	No	No	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	Date	No	No	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	No	No	
<b>interval</b>	Boolean	No	No	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	Integer	No	No	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	Boolean	No	No	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	String	No	No	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	String	No	No	The ID of the object which this is attached to.

Field Name	Type	POST	PUT	Description
<b>objectType</b>	<b>String</b>	No	No	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	No	No	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	No	No	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	No	No	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	No	No	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	No	No	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.

Field Name	Type	POST	PUT	Description
<b>threshold</b>	<b>String</b>	No	No	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	No	No	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	No	No	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.

Value	Description
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.

Value	Description
<b>SOFT_KILL</b>	<p>By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).</p> <p>NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.</p>

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Related topics

- [Reservations on page 1646](#)

Fields: Resource Types

 See the associated [Resource types on page 1654](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	resource-types	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	resource-types.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>

Type	Value	Additional information
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

API version 3

[ResourceType](#)

Represents a resource type in Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this resource type.

API version 2

[ResourceType](#)

Represents a resource type in Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this resource type.

API version 1

[ResourceType](#)

Represents a resource type in Moab Workload Manager.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this resource type.

Related topics

- [Resource types on page 1654](#)

Fields: Roles

 See the associated [Roles on page 1655](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	roles	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	roles.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

API version 3

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.

Field Name	Type	POST	PUT	Description
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#"><u>PrivilegeScope</u></a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.

Value	Description
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

API version 2

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.

Field Name	Type	POST	PUT	Description
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#"><u>PrivilegeScope</u></a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.

Value	Description
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

## API version 1

Role

A role defines a set of permissions that are based on the proxy-user. If no proxy user is specified then access to objects in MWS are limited to its application permissions. For example if the application has permission to update all resources in MWS and no proxy-user is specified in the request then the request can access all resources in MWS.

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>description</b>	<b>String</b>	Yes	Yes	The role description.
<b>name</b>	<b>String</b>	Yes	Yes	The unique human-readable name of this role. Required during POST.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	Yes	Yes	The set of permissions enforced based on the proxy-user.
<b>scope</b>	<a href="#">PrivilegeScope</a>	No	No	

Permission

Represents a permission

Field Name	Type	POST	PUT	Description
<b>id</b>	<b>String</b>	No	No	The unique ID of this role.
<b>action</b>	<b>String</b>	No	No	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	No	No	A description of this permission.
<b>fieldPath</b>	<b>String</b>	No	No	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	No	No	A human readable label for this permission.

Field Name	Type	POST	PUT	Description
<b>resource</b>	<b>String</b>	No	No	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	No	No	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#"><u>PrivilegeScope</u></a>	No	No	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	No	No	The type of the permission. Only 'api' type permissions are enforced.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

[PrivilegeScope](#)

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.

Value	Description
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

Related topics

- [Roles on page 1655](#)

## Fields: Report Samples

 See the associated [Reports on page 1636](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	reports/samples	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	reports.samples.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

Sample

A single snapshot of system state. It can contain all the same information as [Datapoint.data](#) in the sample's [data](#) field.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>agent</b>	<b>String</b>	No	A unique identifier for the agent that recorded this sample.
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	Arbitrary data that was recorded for this sample. Defaults to an empty object if none is supplied.
<b>timestamp</b>	<b>Date</b>	No	The date and time at which this sample was recorded. Defaults to the current date if none is supplied.

## API version 2

Sample

A single snapshot of system state. It can contain all the same information as [Datapoint.data](#) in the sample's [data](#) field.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>agent</b>	<b>String</b>	No	A unique identifier for the agent that recorded this sample.
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	Arbitrary data that was recorded for this sample. Defaults to an empty object if none is supplied.
<b>timestamp</b>	<b>Date</b>	No	The date and time at which this sample was recorded. Defaults to the current date if none is supplied.

## API version 1

Sample

A single snapshot of system state. It can contain all the same information as [Datapoint.data](#) in the sample's [data](#) field.

Field Name	Type	POST	Description
<b>id</b>	<b>Long</b>	No	
<b>agent</b>	<b>String</b>	No	A unique identifier for the agent that recorded this sample.
<b>data</b>	<b>Map&lt;String, Map&gt;</b>	No	Arbitrary data that was recorded for this sample. Defaults to an empty object if none is supplied.
<b>timestamp</b>	<b>Date</b>	No	The date and time at which this sample was recorded. Defaults to the current date if none is supplied.

## Related topics

- [Reports on page 1636](#)

## Fields: Standing Reservations

**i** See the associated [Standing reservations on page 1661](#) resource section for more information on how to use this resource and supported operations.

Additional references

Type	Value	Additional information
<b>Permissions resource</b>	standing-reservations	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	standing-reservations.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	No	<a href="#">Distinct on page 1526</a>

## API version 3

StandingReservation

This class represents a standing reservation.

A standing reservation is any reservation that is not a one-time reservation. This includes reservations that recur every day or every week, or infinite reservations.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the standing reservation.
<b>access</b>	<a href="#">ReservationAccess</a>	If set to <a href="#">ReservationAccess.SHARED</a> , allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.
<b>accounts</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated accounts may use the resources contained within this reservation.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	The set of access control rules associated with this standing reservation.
<b>chargeAccount</b>	<b>String</b>	Specifies the account to which Moab will charge all idle cycles within the reservation (via the allocation manager).
<b>chargeUser</b>	<b>String</b>	Specifies the user to which Moab will charge all idle cycles within the reservation (via the allocation manager). Must be used in conjunction with <a href="#">chargeAccount</a>
<b>classes</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated classes/queues may use the resources contained within this reservation.
<b>clusters</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs originating within the listed clusters may use the resources contained within this reservation.
<b>comment</b>	<b>String</b>	Specifies a descriptive message associated with the standing reservation and all child reservations

Field Name	Type	Description
<b>days</b>	<b>Set&lt;String&gt;</b>	Specifies which days of the week the standing reservation is active. Valid values are Mon, Tue, Wed, Thu, Fri, Sat, Sun, or [ALL].
<b>depth</b>	<b>Integer</b>	Specifies the depth of standing reservations to be created, starting at depth 0 (one per <a href="#">period</a> ).
<b>disabled</b>	<b>Boolean</b>	Specifies if the standing reservation should no longer spawn child reservations.
<b>endOffset</b>	<b>Long</b>	The ending offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. See examples at <a href="#">startOffset</a> .
<b>flags</b>	<a href="#">Set&lt;ReservationFlag&gt;</a>	Specifies special reservation attributes.
<b>groups</b>	<b>Set&lt;String&gt;</b>	Specifies the groups allowed access to this standing reservation.
<b>hosts</b>	<b>Set&lt;String&gt;</b>	Specifies the set of hosts that the scheduler can search for resources to satisfy the reservation. If specified using the class:X format, Moab only selects hosts that support the specified class. If TASKCOUNT is also specified, only TASKCOUNT tasks are reserved. Otherwise, all matching hosts are reserved.
<b>jobAttributes</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Specifies job attributes that grant a job access to the reservation. Values can be specified with a != assignment to only allow jobs NOT requesting a certain feature inside the reservation.
<b>maxJob</b>	<b>Integer</b>	Specifies the maximum number of jobs that can run in the reservation.
<b>maxTime</b>	<b>Integer</b>	Specifies the maximum time for jobs allowable. Can be used with affinity to attract jobs with same maxTime.
<b>messages</b>	<b>Set&lt;String&gt;</b>	Messages associated with the reservation.

Field Name	Type	Description
<b>nodeFeatures</b>	<b>Set&lt;String&gt;</b>	Specifies the required node features for nodes that are part of the standing reservation.
<b>os</b>	<b>String</b>	Specifies the operating system that should be in place during the reservation. Moab provisions this OS at reservation start and restores the original OS at reservation completion.
<b>owner</b>	<a href="#"><u>EmbeddedCredential</u></a>	Specifies the owner of the reservation. Setting ownership for a reservation grants the user management privileges, including the power to release it. Setting a user as the owner of a reservation gives that user privileges to query and release the reservation. For sandbox reservations, sandboxes are applied to a specific peer only if owner is set to CLUSTER:<PEERNAME>
<b>partition</b>	<b>String</b>	Specifies the partition in which to create the standing reservation. Defaults to ALL.
<b>period</b>	<a href="#"><u>TimeWindow</u></a>	Period of the Standing reservation. Defaults to <a href="#"><u>TimeWindow.DAY</u></a> .
<b>procLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor limit for jobs requesting access to this standing reservation.
<b>psLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor-second limit for jobs requesting access to this standing reservation.
<b>qoses</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the listed QoS names can access the reserved resources.
<b>reservationAccessList</b>	<a href="#"><u>Set&lt;Reservation&gt;</u></a>	A list of reservations to which the specified reservation has access.
<b>reservationGroup</b>	<b>String</b>	The group of the reservation.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	<p>Specifies what resources constitute a single standing reservation task. (Each task must be able to obtain all of its resources as an atomic unit on a single node.) Supported resources currently include the following:</p> <ul style="list-style-type: none"> <li>• PROCS (number of processors)</li> <li>• MEM (real memory in MB)</li> <li>• DISK (local disk in MB)</li> <li>• SWAP (virtual memory in MB)</li> </ul>
<b>rollbackOffset</b>	<b>Integer</b>	<p>Specifies the minimum time in the future at which the reservation may start. This offset is rolling meaning the start time of the reservation will continuously roll back into the future to maintain this offset. Rollback offsets are a good way of providing guaranteed resource access to users under the conditions that they must commit their resources in the future or lose dedicated access. See QoS Credential in the Moab Workload Manager documentation for more information on quality of service and service level agreements.</p>
<b>startOffset</b>	<b>Long</b>	<p>The starting offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. If period is DAY, the offset is from midnight (00:00) of the current day. If period is WEEK, the offset is from midnight Sunday of the current week.</p> <p>Example 1: For a standing reservation that begins at 9:00 and ends at 17:00 every day, period is DAY, startOffset is 32400 (9*60*60), and endOffset is 61200 (17*60*60).</p> <p>Example 2: For a standing reservation that begins at 9:00 Monday and ends at 17:00 Friday every week, period is WEEK, startOffset is 118800 ((24+9)*60*60), and endOffset is 493200 (((5*24)+17)*60*60).</p>

Field Name	Type	Description
<b>taskCount</b>	<b>Integer</b>	Specifies how many tasks should be reserved for the reservation Default is 0 (unlimited tasks).
<b>tasksPerNode</b>	<b>Integer</b>	Specifies the minimum number of tasks per node that must be available on eligible nodes. Default is 0 (no TPN constraint)
<b>timeLimit</b>	<b>Integer</b>	Specifies the maximum allowed overlap between the standing reservation and a job requesting resource access. Default is null (-1 in moab)
<b>triggers</b>	<a href="#">Set&lt;Trigger&gt;</a>	Triggers associated with the reservation.
<b>type</b>	<b>String</b>	The type of the reservation.
<b>users</b>	<b>Set&lt;String&gt;</b>	Specifies which users have access to the resources reserved by this reservation.

### ReservationAccess

The access type of a standing reservation. If set to SHARED, allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.

Value	Description
<b>DEDICATED</b>	
<b>SHARED</b>	

### AclRule

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	Description
<b>affinity</b>	<a href="#"><u>AclAffinity</u></a>	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#"><u>AclAffinity.POSITIVE</u></a> .
<b>comparator</b>	<a href="#"><u>ComparisonOperator</u></a>	The type of comparison to make against the ACL object.  Defaults to <a href="#"><u>ComparisonOperator.EQUAL</u></a> .
<b>type</b>	<a href="#"><u>AclType</u></a>	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.

Value	Description
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User

Value	Description
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported

Value	Description
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

### ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOverlap</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.
<b>NOVMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).

Value	Description
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER>=ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRED</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.

Value	Description
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.

Value	Description
<b>TRIGHASFIRE</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

### JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.

Value	Description
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.

Value	Description
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESSUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

[EmbeddedCredential](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	
<b>type</b>	<a href="#"><u>CredentialType</u></a>	

[CredentialType](#)

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

[TimeWindow](#)

This enumeration represents some common time windows. It can be used when for many purposes, but was created specifically for statistics.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Value	Description
<b>YEAR</b>	
<b>INFINITY</b>	

IntLimit

Field Name	Type	Description
<b>qualifier</b>	<b>String</b>	One of: <ul style="list-style-type: none"> <li>• &lt;</li> <li>• &lt;=</li> <li>• ==</li> <li>• &gt;=</li> <li>• &gt;</li> </ul>
<b>value</b>	<b>Integer</b>	

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Accountable Group.

Field Name	Type	Description
<b>accountingQOS</b>	<b>String</b>	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Accountable User.
<b>aclRules</b>	<a href="#"><u>Set&lt;AclRule&gt;</u></a>	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#"><u>Set&lt;DomainProxyVersion1&gt;</u></a>	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Reservation's comments or description.
<b>creationDate</b>	<b>Date</b>	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	Global reservation ID.

Field Name	Type	Description
<b>hostListExpression</b>	<b>String</b>	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	Messages for the reservation.
<b>owner</b>	<a href="#">EmbeddedCredential</a>	The owner of the reservation
<b>partitionId</b>	<b>String</b>	The ID of the partition this reservation is for.
<b>profile</b>	<b>String</b>	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	The reservation's requirements.
<b>reservationGroup</b>	<b>String</b>	The reservation group to which the reservation belongs.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	<b>Date</b>	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#"><u>ReservationStatistics</u></a>	The reservation's statistical information.
<b>subType</b>	<b>String</b>	The reservation sub-type.
<b>taskCount</b>	<b>Integer</b>	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	The set of variables for this reservation.

[DomainProxyVersion1](#)

Field Name	Type	Description
<b>id</b>	<b>String</b>	The id of the object.

[MessageVersion1](#)

Field Name	Type	Description
<b>author</b>	<b>String</b>	The author of the message.
<b>creationTime</b>	<b>Date</b>	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	An optional priority that can be attached to the comment.

#### ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	Required feature mode.
<b>memory</b>	<b>Integer</b>	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	The list of node IDs required for this reservation.
<b>os</b>	<b>String</b>	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Required task count.

ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	Description
<b>caps</b>	<b>Double</b>	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	The total idle processor-seconds over the life of the reservation.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#">TriggerActionType</a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#">TriggerEventType</a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

### TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

### TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers

Field Name	Type	Description
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#">TriggerActionType</a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#">TriggerEventType</a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.

Field Name	Type	Description
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

## API version 2

StandingReservation

This class represents a standing reservation.

A standing reservation is any reservation that is not a one-time reservation. This includes reservations that recur every day or every week, or infinite reservations.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the standing reservation.
<b>access</b>	<a href="#">ReservationAccess</a>	If set to <a href="#">ReservationAccess.SHARED</a> , allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.
<b>accounts</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated accounts may use the resources contained within this reservation.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	The set of access control rules associated with this standing reservation.
<b>chargeAccount</b>	<b>String</b>	Specifies the account to which Moab will charge all idle cycles within the reservation (via the allocation manager).
<b>chargeUser</b>	<b>String</b>	Specifies the user to which Moab will charge all idle cycles within the reservation (via the allocation manager). Must be used in conjunction with <a href="#">chargeAccount</a>
<b>classes</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated classes/queues may use the resources contained within this reservation.
<b>clusters</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs originating within the listed clusters may use the resources contained within this reservation.
<b>comment</b>	<b>String</b>	Specifies a descriptive message associated with the standing reservation and all child reservations

Field Name	Type	Description
<b>days</b>	<b>Set&lt;String&gt;</b>	Specifies which days of the week the standing reservation is active. Valid values are Mon, Tue, Wed, Thu, Fri, Sat, Sun, or [ALL].
<b>depth</b>	<b>Integer</b>	Specifies the depth of standing reservations to be created, starting at depth 0 (one per <a href="#">period</a> ).
<b>disabled</b>	<b>Boolean</b>	Specifies if the standing reservation should no longer spawn child reservations.
<b>endOffset</b>	<b>Long</b>	The ending offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. See examples at <a href="#">startOffset</a> .
<b>flags</b>	<a href="#">Set&lt;ReservationFlag&gt;</a>	Specifies special reservation attributes.
<b>groups</b>	<b>Set&lt;String&gt;</b>	Specifies the groups allowed access to this standing reservation.
<b>hosts</b>	<b>Set&lt;String&gt;</b>	Specifies the set of hosts that the scheduler can search for resources to satisfy the reservation. If specified using the class:X format, Moab only selects hosts that support the specified class. If TASKCOUNT is also specified, only TASKCOUNT tasks are reserved. Otherwise, all matching hosts are reserved.
<b>jobAttributes</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Specifies job attributes that grant a job access to the reservation. Values can be specified with a != assignment to only allow jobs NOT requesting a certain feature inside the reservation.
<b>maxJob</b>	<b>Integer</b>	Specifies the maximum number of jobs that can run in the reservation.
<b>maxTime</b>	<b>Integer</b>	Specifies the maximum time for jobs allowable. Can be used with affinity to attract jobs with same maxTime.
<b>messages</b>	<b>Set&lt;String&gt;</b>	Messages associated with the reservation.

Field Name	Type	Description
<b>nodeFeatures</b>	<b>Set&lt;String&gt;</b>	Specifies the required node features for nodes that are part of the standing reservation.
<b>os</b>	<b>String</b>	Specifies the operating system that should be in place during the reservation. Moab provisions this OS at reservation start and restores the original OS at reservation completion.
<b>owner</b>	<a href="#"><u>EmbeddedCredential</u></a>	Specifies the owner of the reservation. Setting ownership for a reservation grants the user management privileges, including the power to release it. Setting a user as the owner of a reservation gives that user privileges to query and release the reservation. For sandbox reservations, sandboxes are applied to a specific peer only if owner is set to CLUSTER:<PEERNAME>
<b>partition</b>	<b>String</b>	Specifies the partition in which to create the standing reservation. Defaults to ALL.
<b>period</b>	<a href="#"><u>TimeWindow</u></a>	Period of the Standing reservation. Defaults to <a href="#"><u>TimeWindow.DAY</u></a> .
<b>procLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor limit for jobs requesting access to this standing reservation.
<b>psLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor-second limit for jobs requesting access to this standing reservation.
<b>qoses</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the listed QoS names can access the reserved resources.
<b>reservationAccessList</b>	<a href="#"><u>Set&lt;Reservation&gt;</u></a>	A list of reservations to which the specified reservation has access.
<b>reservationGroup</b>	<b>String</b>	The group of the reservation.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	<p>Specifies what resources constitute a single standing reservation task. (Each task must be able to obtain all of its resources as an atomic unit on a single node.) Supported resources currently include the following:</p> <ul style="list-style-type: none"> <li>• PROCS (number of processors)</li> <li>• MEM (real memory in MB)</li> <li>• DISK (local disk in MB)</li> <li>• SWAP (virtual memory in MB)</li> </ul>
<b>rollbackOffset</b>	<b>Integer</b>	<p>Specifies the minimum time in the future at which the reservation may start. This offset is rolling meaning the start time of the reservation will continuously roll back into the future to maintain this offset. Rollback offsets are a good way of providing guaranteed resource access to users under the conditions that they must commit their resources in the future or lose dedicated access. See QoS Credential in the Moab Workload Manager documentation for more information on quality of service and service level agreements.</p>
<b>startOffset</b>	<b>Long</b>	<p>The starting offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. If period is DAY, the offset is from midnight (00:00) of the current day. If period is WEEK, the offset is from midnight Sunday of the current week.</p> <p>Example 1: For a standing reservation that begins at 9:00 and ends at 17:00 every day, period is DAY, startOffset is 32400 (9*60*60), and endOffset is 61200 (17*60*60).</p> <p>Example 2: For a standing reservation that begins at 9:00 Monday and ends at 17:00 Friday every week, period is WEEK, startOffset is 118800 ((24+9)*60*60), and endOffset is 493200 (((5*24)+17)*60*60).</p>

Field Name	Type	Description
<b>taskCount</b>	<b>Integer</b>	Specifies how many tasks should be reserved for the reservation Default is 0 (unlimited tasks).
<b>tasksPerNode</b>	<b>Integer</b>	Specifies the minimum number of tasks per node that must be available on eligible nodes. Default is 0 (no TPN constraint)
<b>timeLimit</b>	<b>Integer</b>	Specifies the maximum allowed overlap between the standing reservation and a job requesting resource access. Default is null (-1 in moab)
<b>triggers</b>	<a href="#">Set&lt;Trigger&gt;</a>	Triggers associated with the reservation.
<b>type</b>	<b>String</b>	The type of the reservation.
<b>users</b>	<b>Set&lt;String&gt;</b>	Specifies which users have access to the resources reserved by this reservation.

### ReservationAccess

The access type of a standing reservation. If set to SHARED, allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.

Value	Description
<b>DEDICATED</b>	
<b>SHARED</b>	

### AclRule

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	Description
<b>affinity</b>	<a href="#"><u>AclAffinity</u></a>	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#"><u>AclAffinity.POSITIVE</u></a> .
<b>comparator</b>	<a href="#"><u>ComparisonOperator</u></a>	The type of comparison to make against the ACL object.  Defaults to <a href="#"><u>ComparisonOperator.EQUAL</u></a> .
<b>type</b>	<a href="#"><u>AclType</u></a>	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.

Value	Description
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User

Value	Description
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JPRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported

Value	Description
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

### ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOverlap</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.
<b>NOVMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).

Value	Description
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER>=ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRED</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.

Value	Description
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.

Value	Description
<b>TRIGHASFIRE</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

### JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.

Value	Description
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.

Value	Description
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESSUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

[EmbeddedCredential](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	
<b>type</b>	<a href="#"><u>CredentialType</u></a>	

[CredentialType](#)

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

[TimeWindow](#)

This enumeration represents some common time windows. It can be used when for many purposes, but was created specifically for statistics.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Value	Description
<b>YEAR</b>	
<b>INFINITY</b>	

IntLimit

Field Name	Type	Description
<b>qualifier</b>	<b>String</b>	One of: <ul style="list-style-type: none"> <li>• &lt;</li> <li>• &lt;=</li> <li>• ==</li> <li>• &gt;=</li> <li>• &gt;</li> </ul>
<b>value</b>	<b>Integer</b>	

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Accountable Group.

Field Name	Type	Description
<b>accountingQOS</b>	<b>String</b>	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Accountable User.
<b>aclRules</b>	<a href="#"><u>Set&lt;AclRule&gt;</u></a>	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#"><u>Set&lt;DomainProxyVersion1&gt;</u></a>	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Reservation's comments or description.
<b>creationDate</b>	<b>Date</b>	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	Global reservation ID.

Field Name	Type	Description
<b>hostListExpression</b>	<b>String</b>	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	Messages for the reservation.
<b>owner</b>	<a href="#">EmbeddedCredential</a>	The owner of the reservation
<b>partitionId</b>	<b>String</b>	The ID of the partition this reservation is for.
<b>profile</b>	<b>String</b>	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	The reservation's requirements.
<b>reservationGroup</b>	<b>String</b>	The reservation group to which the reservation belongs.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	<b>Date</b>	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#"><u>ReservationStatistics</u></a>	The reservation's statistical information.
<b>subType</b>	<b>String</b>	The reservation sub-type.
<b>taskCount</b>	<b>Integer</b>	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	The set of variables for this reservation.

[DomainProxyVersion1](#)

Field Name	Type	Description
<b>id</b>	<b>String</b>	The id of the object.

[MessageVersion1](#)

Field Name	Type	Description
<b>author</b>	<b>String</b>	The author of the message.
<b>creationTime</b>	<b>Date</b>	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	An optional priority that can be attached to the comment.

#### ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	Required feature mode.
<b>memory</b>	<b>Integer</b>	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	The list of node IDs required for this reservation.
<b>os</b>	<b>String</b>	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Required task count.

ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	Description
<b>caps</b>	<b>Double</b>	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	The total idle processor-seconds over the life of the reservation.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#">TriggerActionType</a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#">TriggerEventType</a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

### TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers

Field Name	Type	Description
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.

Field Name	Type	Description
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

## API version 1

StandingReservation

This class represents a standing reservation.

A standing reservation is any reservation that is not a one-time reservation. This includes reservations that recur every day or every week, or infinite reservations.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the standing reservation.
<b>access</b>	<a href="#">ReservationAccess</a>	If set to <a href="#">ReservationAccess.SHARED</a> , allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.
<b>accounts</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated accounts may use the resources contained within this reservation.
<b>aclRules</b>	<a href="#">Set&lt;AclRule&gt;</a>	The set of access control rules associated with this standing reservation.
<b>chargeAccount</b>	<b>String</b>	Specifies the account to which Moab will charge all idle cycles within the reservation (via the allocation manager).
<b>chargeUser</b>	<b>String</b>	Specifies the user to which Moab will charge all idle cycles within the reservation (via the allocation manager). Must be used in conjunction with <a href="#">chargeAccount</a>
<b>classes</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the associated classes/queues may use the resources contained within this reservation.
<b>clusters</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs originating within the listed clusters may use the resources contained within this reservation.
<b>comment</b>	<b>String</b>	Specifies a descriptive message associated with the standing reservation and all child reservations

Field Name	Type	Description
<b>days</b>	<b>Set&lt;String&gt;</b>	Specifies which days of the week the standing reservation is active. Valid values are Mon, Tue, Wed, Thu, Fri, Sat, Sun, or [ALL].
<b>depth</b>	<b>Integer</b>	Specifies the depth of standing reservations to be created, starting at depth 0 (one per <a href="#">period</a> ).
<b>disabled</b>	<b>Boolean</b>	Specifies if the standing reservation should no longer spawn child reservations.
<b>endOffset</b>	<b>Long</b>	The ending offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. See examples at <a href="#">startOffset</a> .
<b>flags</b>	<a href="#">Set&lt;ReservationFlag&gt;</a>	Specifies special reservation attributes.
<b>groups</b>	<b>Set&lt;String&gt;</b>	Specifies the groups allowed access to this standing reservation.
<b>hosts</b>	<b>Set&lt;String&gt;</b>	Specifies the set of hosts that the scheduler can search for resources to satisfy the reservation. If specified using the class:X format, Moab only selects hosts that support the specified class. If TASKCOUNT is also specified, only TASKCOUNT tasks are reserved. Otherwise, all matching hosts are reserved.
<b>jobAttributes</b>	<a href="#">Set&lt;JobFlag&gt;</a>	Specifies job attributes that grant a job access to the reservation. Values can be specified with a != assignment to only allow jobs NOT requesting a certain feature inside the reservation.
<b>maxJob</b>	<b>Integer</b>	Specifies the maximum number of jobs that can run in the reservation.
<b>maxTime</b>	<b>Integer</b>	Specifies the maximum time for jobs allowable. Can be used with affinity to attract jobs with same maxTime.
<b>messages</b>	<b>Set&lt;String&gt;</b>	Messages associated with the reservation.

Field Name	Type	Description
<b>nodeFeatures</b>	<b>Set&lt;String&gt;</b>	Specifies the required node features for nodes that are part of the standing reservation.
<b>os</b>	<b>String</b>	Specifies the operating system that should be in place during the reservation. Moab provisions this OS at reservation start and restores the original OS at reservation completion.
<b>owner</b>	<a href="#"><u>EmbeddedCredential</u></a>	Specifies the owner of the reservation. Setting ownership for a reservation grants the user management privileges, including the power to release it. Setting a user as the owner of a reservation gives that user privileges to query and release the reservation. For sandbox reservations, sandboxes are applied to a specific peer only if owner is set to CLUSTER:<PEERNAME>
<b>partition</b>	<b>String</b>	Specifies the partition in which to create the standing reservation. Defaults to ALL.
<b>period</b>	<a href="#"><u>TimeWindow</u></a>	Period of the Standing reservation. Defaults to <a href="#"><u>TimeWindow.DAY</u></a> .
<b>procLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor limit for jobs requesting access to this standing reservation.
<b>psLimit</b>	<a href="#"><u>IntLimit</u></a>	Specifies the processor-second limit for jobs requesting access to this standing reservation.
<b>qoses</b>	<b>Set&lt;String&gt;</b>	Specifies that jobs with the listed QoS names can access the reserved resources.
<b>reservationAccessList</b>	<a href="#"><u>Set&lt;Reservation&gt;</u></a>	A list of reservations to which the specified reservation has access.
<b>reservationGroup</b>	<b>String</b>	The group of the reservation.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	<p>Specifies what resources constitute a single standing reservation task. (Each task must be able to obtain all of its resources as an atomic unit on a single node.) Supported resources currently include the following:</p> <ul style="list-style-type: none"> <li>• PROCS (number of processors)</li> <li>• MEM (real memory in MB)</li> <li>• DISK (local disk in MB)</li> <li>• SWAP (virtual memory in MB)</li> </ul>
<b>rollbackOffset</b>	<b>Integer</b>	<p>Specifies the minimum time in the future at which the reservation may start. This offset is rolling meaning the start time of the reservation will continuously roll back into the future to maintain this offset. Rollback offsets are a good way of providing guaranteed resource access to users under the conditions that they must commit their resources in the future or lose dedicated access. See QoS Credential in the Moab Workload Manager documentation for more information on quality of service and service level agreements.</p>
<b>startOffset</b>	<b>Long</b>	<p>The starting offset, in seconds, from the beginning of the current <a href="#">period</a> (DAY or WEEK), for this standing reservation. If period is DAY, the offset is from midnight (00:00) of the current day. If period is WEEK, the offset is from midnight Sunday of the current week.</p> <p>Example 1: For a standing reservation that begins at 9:00 and ends at 17:00 every day, period is DAY, startOffset is 32400 (9*60*60), and endOffset is 61200 (17*60*60).</p> <p>Example 2: For a standing reservation that begins at 9:00 Monday and ends at 17:00 Friday every week, period is WEEK, startOffset is 118800 ((24+9)*60*60), and endOffset is 493200 (((5*24)+17)*60*60).</p>

Field Name	Type	Description
<b>taskCount</b>	<b>Integer</b>	Specifies how many tasks should be reserved for the reservation Default is 0 (unlimited tasks).
<b>tasksPerNode</b>	<b>Integer</b>	Specifies the minimum number of tasks per node that must be available on eligible nodes. Default is 0 (no TPN constraint)
<b>timeLimit</b>	<b>Integer</b>	Specifies the maximum allowed overlap between the standing reservation and a job requesting resource access. Default is null (-1 in moab)
<b>triggers</b>	<a href="#">Set&lt;Trigger&gt;</a>	Triggers associated with the reservation.
<b>type</b>	<b>String</b>	The type of the reservation.
<b>users</b>	<b>Set&lt;String&gt;</b>	Specifies which users have access to the resources reserved by this reservation.

### [ReservationAccess](#)

The access type of a standing reservation. If set to SHARED, allows a standing reservation to use resources already allocated to other non-job reservations. Otherwise, these other reservations block resource access.

Value	Description
<b>DEDICATED</b>	
<b>SHARED</b>	

### [AclRule](#)

This class represents a rule that can be in Moab's access control list (ACL) mechanism.

The basic AclRule information is the object's name and type. The type directly maps to an [AclType](#) value. The default mechanism Moab uses to check the ACL for a particular item is if the user or object coming in has ANY of the values in the ACL, then the user or object is given access. If no values match the user or object in question, the user or object is rejected access.

Field Name	Type	Description
<b>affinity</b>	<a href="#"><u>AclAffinity</u></a>	Reservation ACLs allow or deny access to reserved resources but they may also be configured to affect a job's affinity for a particular reservation. By default, jobs gravitate toward reservations through a mechanism known as positive affinity. This mechanism allows jobs to run on the most constrained resources leaving other, unreserved resources free for use by other jobs that may not be able to access the reserved resources. Normally this is a desired behavior. However, sometimes, it is desirable to reserve resources for use only as a last resort-using the reserved resources only when there are no other resources available. This last resort behavior is known as negative affinity.  Defaults to <a href="#"><u>AclAffinity.POSITIVE</u></a> .
<b>comparator</b>	<a href="#"><u>ComparisonOperator</u></a>	The type of comparison to make against the ACL object.  Defaults to <a href="#"><u>ComparisonOperator.EQUAL</u></a> .
<b>type</b>	<a href="#"><u>AclType</u></a>	The type of the object that is being granted (or denied) access.
<b>value</b>	<b>String</b>	The name of the object that is being granted (or denied) access.

[AclAffinity](#)

This enumeration describes the values available for describing how a rule is used in establishing access to an object in Moab. Currently, these ACL affinities are used only for granting access to reservations.

Value	Description
<b>NEGATIVE</b>	Access to the object is repelled using this rule until access is the last choice.
<b>NEUTRAL</b>	Access to the object is not affected by affinity.
<b>POSITIVE</b>	Access to the object is looked at as the first choice.
<b>PREEMPTIBLE</b>	Access to the object given the rule gives preemptible status to the accessor. Supported only during GET.

Value	Description
<b>REQUIRED</b>	The rule in question must be satisfied in order to gain access to the object. Supported only during GET.
<b>UNAVAILABLE</b>	The rule does not have its affinity available. Supported only during GET.

### ComparisonOperator

This enumeration is used when Moab needs to compare items. One such use is in Access Control Lists (ACLs).

Value	Description
<b>GREATER_THAN</b>	Valid values: ">", "gt"
<b>GREATER_THAN_OR_EQUAL</b>	Valid values: ">=", "ge"
<b>LESS_THAN</b>	Valid values: "<", "lt"
<b>LESS_THAN_OR_EQUAL</b>	Valid values: "<=", "le"
<b>EQUAL</b>	Valid values: "==", "eq", "="
<b>NOT_EQUAL</b>	Valid values: "!=", "ne", "<>"
<b>LEXIGRAPHIC_SUBSTRING</b>	Valid value: "%<"
<b>LEXIGRAPHIC_NOT_EQUAL</b>	Valid value: "%!"
<b>LEXIGRAPHIC_EQUAL</b>	Valid value: "%="

### AcType

This enumeration describes the values available for the type of an ACL Rule.

Value	Description
<b>USER</b>	User

Value	Description
<b>GROUP</b>	Group
<b>ACCOUNT</b>	Account or Project
<b>CLASS</b>	Class or Queue
<b>QOS</b>	Quality of Service
<b>CLUSTER</b>	Cluster
<b>JOB_ID</b>	Job ID
<b>RESERVATION_ID</b>	Reservation ID
<b>JOB_TEMPLATE</b>	Job Template
<b>JOB_ATTRIBUTE</b>	Job Attribute
<b>DURATION</b>	Duration in Seconds
<b>PROCESSOR_SECONDS</b>	Processor Seconds
<b>JRIORITY</b>	Not supported
<b>MEMORY</b>	Not supported
<b>NODE</b>	Not supported
<b>PAR</b>	Not supported
<b>PROC</b>	Not supported
<b>QTIME</b>	Not supported
<b>QUEUE</b>	Not supported
<b>RACK</b>	Not supported

Value	Description
<b>SCHED</b>	Not supported
<b>SYSTEM</b>	Not supported
<b>TASK</b>	Not supported
<b>VC</b>	Not supported
<b>XFACTOR</b>	Not supported

### ReservationFlag

The flag types of a reservation.

Value	Description
<b>ALLOWJOBOverlap</b>	Allows jobs to overlap this Reservation, but not start during it (unless they have ACL access).
<b>APPLYPROFRESOURCES</b>	Only apply resource allocation info from profile.
<b>DEADLINE</b>	Reservation should be scheduled against a deadline.
<b>IGNIDLEJOBS</b>	Ignore idle job reservations.
<b>IGNJOBRSV</b>	Ignore job reservations, but not user or other reservations.
<b>CHARGE</b>	Charge the idle cycles in the accounting manager.
<b>NOVMIGRATIONS</b>	Override the VM Migration Policy and don't migrate VMs that overlap this reservation.
<b>OWNERPREEMPTIGNOREMINTIME</b>	Owner ignores preemptmintime for this reservation.
<b>PROVISION</b>	Reservation should be capable of provisioning.
<b>NOACLOVERLAP</b>	Reservation will not look at ACLs to overlap job (when using exclusive).

Value	Description
<b>ADVRES</b>	If set, the reservation is created in advance of needing it.
<b>ADVRESJOBDESTROY</b>	Cancel any jobs associated with the reservation when it is released.
<b>ALLOWGRID</b>	The reservation is set up for use in a grid environment.
<b>ALLOWPRSV</b>	Personal reservations can be created within the space of this standing reservation (and ONLY this standing reservation). By default, when a standing reservation is given the flag ALLOWPRSV, it is given the ACL rule USER>=ALL+ allowing all jobs and all users access.
<b>BYNAME</b>	Reservation only allows access to jobs that meet reservation ACLs and explicitly request the resources of this reservation using the job ADVRES flag.
<b>DEDICATEDNODE</b>	If set, only one active reservation is allowed on a node.
<b>DEDICATEDRESOURCE</b>	The reservation is only placed on resources that are not reserved by any other reservation, including jobs and other reservations.
<b>EXCLUDEJOBS</b>	Makes a reservation job exclusive, where only one job can run in the reservation.
<b>ENDTRIGHASFIRED</b>	A trigger has finished firing.
<b>ENFORCENODESET</b>	Enforce node sets when creating reservation.
<b>EXCLUDEALLBUTSB</b>	Reservation only shares resources with sandboxes.
<b>EXCLUDEMYGROUP</b>	Exclude reservations within the same group.
<b>IGNRSV</b>	Forces the reservation onto nodes regardless of whether there are other reservations currently residing on the nodes.
<b>IGNSTATE</b>	Request ignores existing resource reservations, allowing the reservation to be forced onto available resources even if this conflicts with other reservations.

Value	Description
<b>ISACTIVE</b>	If set, the reservation is currently active.
<b>ISCLOSED</b>	If set, the reservation is closed.
<b>ISGLOBAL</b>	If set the reservation applies to all resources.
<b>OWNERPREEMPT</b>	The owner of the reservation is given preemptor status for resources contained in the reservation.
<b>PARENTLOCK</b>	The reservation can only be destroyed by destroying its parent.
<b>PREEMPTEE</b>	The reservation is preemptible.
<b>PLACEHOLDER</b>	The reservation is a placeholder for resources.
<b>PRSV</b>	The reservation is a non-administrator, non-standing reservation, user-created reservation.
<b>REQFULL</b>	The reservation will fail if all resources requested cannot be allocated.
<b>SCHEDULEVCRSV</b>	The reservation was created as part of a schedule VC command. This pertains to reservations creating while scheduling MWS Services, and these are filtered from the MWS output of reservations.
<b>SINGLEUSE</b>	The reservation is automatically removed after completion of the first job to use the reserved resources.
<b>SPACEFLEX</b>	The reservation is allowed to adjust resources allocated over time in an attempt to optimize resource utilization.
<b>STANDINGRSV</b>	If set, the reservation was created by a standing reservation instance.
<b>STATIC</b>	Makes a reservation ineligible to modified or canceled by an administrator.
<b>SYSTEMJOB</b>	The reservation was created by a system job.
<b>TIMEFLEX</b>	The reservation is allowed to adjust the reserved time frame in an attempt to optimize resource utilization.

Value	Description
<b>TRIGHASFIRE</b>	The reservation has one or more triggers that have fired on it.
<b>WASACTIVE</b>	The reservation was previously active.
<b>EVACVMS</b>	Evacuate virtual machines on the node when the reservation starts.

### JobFlag

This enumeration specifies the flag types of a job.

Value	Description
<b>NONE</b>	
<b>BACKFILL</b>	The job is using backfill to run.
<b>COALLOC</b>	The job can use resources from multiple resource managers and partitions.
<b>ADVRES</b>	The job requires the use of a reservation.
<b>NOQUEUE</b>	The job will attempt to execute immediately or fail.
<b>ARRAYJOB</b>	The job is part of a job array.
<b>ARRAYJOBPARLOCK</b>	This array job will only run in one partition.
<b>ARRAYJOBPARSPAN</b>	This array job will span partitions (default).
<b>ARRAYMASTER</b>	This job is the master of a job array.
<b>BESTEFFORT</b>	The job will succeed if even partial resources are available.
<b>RESTARTABLE</b>	The job is restartable.
<b>SUSPENDABLE</b>	The job is suspendable.
<b>HASPREEMPTED</b>	This job preempted other jobs to start.

Value	Description
<b>PREEMPTEE</b>	The job is a preemptee and therefore can be preempted by other jobs.
<b>PREEMPTOR</b>	The job is a preemptor and therefore can preempt other jobs.
<b>RSVMAP</b>	The job is based on a reservation.
<b>SPVIOLATION</b>	The job was started with a soft policy violation.
<b>IGNNODEPOLICIES</b>	The job will ignore node policies.
<b>IGNPOLICIES</b>	The job will ignore idle, active, class, partition, and system policies.
<b>IGNNODESTATE</b>	The job will ignore node state in order to run.
<b>IGNIDLEJOBRSV</b>	The job can ignore idle job reservations. The job granted access to all idle job reservations.
<b>INTERACTIVE</b>	The job needs to interactive input from the user to run.
<b>FSVIOLATION</b>	The job was started with a fairshare violation.
<b>GLOBALQUEUE</b>	The job is directly submitted without doing any authentication.
<b>NORESOURCES</b>	The job is a system job that does not need any resources.
<b>NORMSTART</b>	The job will not query a resource manager to run.
<b>CLUSTERLOCKED</b>	The job is locked into the current cluster and cannot be migrated elsewhere. This is for grid mode.
<b>FRAGMENT</b>	The job can be run across multiple nodes in individual chunks.
<b>SYSTEMJOB</b>	The job is a system job which simply runs on the same node that Moab is running on. This is usually used for running scripts and other executables in workflows.
<b>ADMINSETIGNPOLICIES</b>	The <a href="#">IGNPOLICIES</a> flag was set by an administrator.

Value	Description
<b>EXTENDSTARTWALLTIME</b>	The job duration (walltime) was extended at job start.
<b>SHAREDMEM</b>	The job will share its memory across nodes.
<b>BLOCKEDBYGRES</b>	The job's generic resource requirement caused the job to start later.
<b>GRESONLY</b>	The job is requesting only generic resources, no compute resources.
<b>TEMPLATESAPPLIED</b>	The job has had all applicable templates applied to it.
<b>META</b>	META job, just a container around resources.
<b>WIDERSVSEARCHALGO</b>	This job prefers the wide search algorithm.
<b>VMTRACKING</b>	The job is a VMTracking job for an externally-created VM (via job template).
<b>DESTROYTEMPLATESSUBMITTED</b>	A destroy job has already been created from the template for this job.
<b>PROCSPECIFIED</b>	The job requested processors on the command line.
<b>CANCELONFIRSTFAILURE</b>	Cancel job array on first array job failure.
<b>CANCELONFIRSTSUCCESS</b>	Cancel job array on first array job success.
<b>CANCELONANYFAILURE</b>	Cancel job array on any array job failure.
<b>CANCELONANYSUCCESS</b>	Cancel job array on any array job success.
<b>CANCELONEXITCODE</b>	Cancel job array on a specific exit code.
<b>NOVMMIGRATE</b>	Do not migrate the virtual machine that this job sets up.
<b>PURGEONSUCCESSONLY</b>	Only purge the job if it completed successfully

[EmbeddedCredential](#)

Field Name	Type	Description
<b>name</b>	<b>String</b>	
<b>type</b>	<a href="#"><u>CredentialType</u></a>	

[CredentialType](#)

Value	Description
<b>USER</b>	
<b>GROUP</b>	
<b>ACCOUNT</b>	
<b>CLASS</b>	
<b>QOS</b>	
<b>NOT_SPECIFIED</b>	

[TimeWindow](#)

This enumeration represents some common time windows. It can be used when for many purposes, but was created specifically for statistics.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Value	Description
<b>YEAR</b>	
<b>INFINITY</b>	

IntLimit

Field Name	Type	Description
<b>qualifier</b>	<b>String</b>	One of: <ul style="list-style-type: none"> <li>• &lt;</li> <li>• &lt;=</li> <li>• ==</li> <li>• &gt;=</li> <li>• &gt;</li> </ul>
<b>value</b>	<b>Integer</b>	

Reservation

A reservation is the mechanism by which Moab guarantees the availability of a set of resources at a particular time. Each reservation consists of three major components: (1) a set of resources, (2) a time frame, and (3) an access control list. It is a scheduler role to ensure that the access control list is not violated during the reservation's lifetime (that is, its time frame) on the resources listed. For example, a reservation may specify that node002 is reserved for user Tom on Friday. The scheduler is thus constrained to make certain that only Tom's jobs can use node002 at any time on Friday.

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the reservation.
<b>accountingAccount</b>	<b>String</b>	Accountable Account.
<b>accountingGroup</b>	<b>String</b>	Accountable Group.

Field Name	Type	Description
<b>accountingQOS</b>	<b>String</b>	Accountable QOS.
<b>accountingUser</b>	<b>String</b>	Accountable User.
<b>aclRules</b>	<a href="#"><u>Set&lt;AclRule&gt;</u></a>	The set of access control rules associated with this reservation.
<b>allocatedNodeCount</b>	<b>Integer</b>	The number of allocated nodes for this reservation.
<b>allocatedNodes</b>	<a href="#"><u>Set&lt;DomainProxyVersion1&gt;</u></a>	The nodes allocated to the reservation.
<b>allocatedProcessorCount</b>	<b>Integer</b>	The number of allocated processors.
<b>allocatedTaskCount</b>	<b>Integer</b>	The number of allocated tasks.
<b>comments</b>	<b>String</b>	Reservation's comments or description.
<b>creationDate</b>	<b>Date</b>	Creation date. Automatically set by Moab when a user creates the reservation.
<b>duration</b>	<b>Long</b>	The duration of the reservation (in seconds).
<b>endDate</b>	<b>Date</b>	The end date of the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation ends.
<b>excludeJobs</b>	<b>Set&lt;String&gt;</b>	The list of jobs to exclude. Client must also set the IGNJOBRSV reservation flag. Otherwise, results are undefined. Used only during reservation creation.
<b>expireDate</b>	<b>Date</b>	The date/time when the reservation expires and vacates.
<b>flags</b>	<a href="#"><u>Set&lt;ReservationFlag&gt;</u></a>	The flags associated with the reservation.
<b>globalId</b>	<b>String</b>	Global reservation ID.

Field Name	Type	Description
<b>hostListExpression</b>	<b>String</b>	The list of nodes a user can select to reserve. This may or may not be the nodes that are currently allocated to this reservation. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>idPrefix</b>	<b>String</b>	The user-specified prefix for this reservation. If provided, Moab combines the idPrefix with an integer, and the combination is the unique identifier for this reservation.
<b>isActive</b>	<b>Boolean</b>	State whether or not this reservation is currently active.
<b>isTracked</b>	<b>Boolean</b>	States whether reservation resource usage is tracked.
<b>label</b>	<b>String</b>	When a label is assigned to a reservation, the reservation can then be referenced by that label as well as by the reservation name.
<b>maxTasks</b>	<b>Integer</b>	The maximum number of tasks for this reservation.
<b>messages</b>	<a href="#">Set&lt;MessageVersion1&gt;</a>	Messages for the reservation.
<b>owner</b>	<a href="#">EmbeddedCredential</a>	The owner of the reservation
<b>partitionId</b>	<b>String</b>	The ID of the partition this reservation is for.
<b>profile</b>	<b>String</b>	The profile that this reservation is using. A profile is a specification of attributes that all reservations share. Used only during reservation creation.
<b>requirements</b>	<a href="#">ReservationRequirement</a>	The reservation's requirements.
<b>reservationGroup</b>	<b>String</b>	The reservation group to which the reservation belongs.

Field Name	Type	Description
<b>resources</b>	<b>Map&lt;String, Integer&gt;</b>	The reservation's resources. This field is a map, where the key is PROCS, MEM DISK, SWAP, or one or more user-defined keys.
<b>startDate</b>	<b>Date</b>	The start time for the reservation. This is especially useful for one-time reservations, which have an exact time for when a reservation starts.
<b>statistics</b>	<a href="#"><u>ReservationStatistics</u></a>	The reservation's statistical information.
<b>subType</b>	<b>String</b>	The reservation sub-type.
<b>taskCount</b>	<b>Integer</b>	The number of tasks that must be allocated to satisfy the reservation request. Note: Either hostListExpression or taskCount must be set to create a reservation.
<b>trigger</b>	<a href="#"><u>Trigger</u></a>	Trigger for reservation. Used only during reservation creation.
<b>triggerIds</b>	<b>Set&lt;String&gt;</b>	The IDs of the triggers attached to this reservation.
<b>uniqueIndex</b>	<b>String</b>	The globally-unique reservation index.
<b>variables</b>	<b>Map&lt;String, Map&gt;</b>	The set of variables for this reservation.

[DomainProxyVersion1](#)

Field Name	Type	Description
<b>id</b>	<b>String</b>	The id of the object.

[MessageVersion1](#)

Field Name	Type	Description
<b>author</b>	<b>String</b>	The author of the message.
<b>creationTime</b>	<b>Date</b>	The time the message was created in epoch time.
<b>expireTime</b>	<b>Date</b>	The time the message will be deleted in epoch time.
<b>index</b>	<b>Integer</b>	The index of the message relative to other messages in Moab's memory.
<b>message</b>	<b>String</b>	The comment information itself.
<b>messageCount</b>	<b>Integer</b>	The number of times this message has been displayed.
<b>priority</b>	<b>Double</b>	An optional priority that can be attached to the comment.

#### ReservationRequirement

Represents all the types of requirements a user can request while creating a reservation.

Field Name	Type	Description
<b>architecture</b>	<b>String</b>	Required architecture.
<b>featureList</b>	<b>Set&lt;String&gt;</b>	The list of features required for this reservation.
<b>featureMode</b>	<b>String</b>	Required feature mode.
<b>memory</b>	<b>Integer</b>	Required node memory, in MB.
<b>nodeCount</b>	<b>Integer</b>	Required number of nodes.
<b>nodeIds</b>	<b>Set&lt;String&gt;</b>	The list of node IDs required for this reservation.
<b>os</b>	<b>String</b>	Required Operating System.
<b>taskCount</b>	<b>Integer</b>	Required task count.

ReservationStatistics

Represents some basic statistical information that is kept about the usage of reservations. All metrics that are kept track relate to processor-seconds usage.

Field Name	Type	Description
<b>caps</b>	<b>Double</b>	The current active processor-seconds in the last reported iteration.
<b>cips</b>	<b>Double</b>	The current idle processor-seconds in the last reported iteration.
<b>taps</b>	<b>Double</b>	The total active processor-seconds over the life of the reservation.
<b>tips</b>	<b>Double</b>	The total idle processor-seconds over the life of the reservation.

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#">TriggerActionType</a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#">TriggerEventType</a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.

Field Name	Type	Description
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#">Set&lt;TriggerFlag&gt;</a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#">TriggerEventType.EPOCH</a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#">TriggerPeriod</a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.

Field Name	Type	Description
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

### TriggerActionType

This enumeration specifies the action type of a trigger.

Value	Description
<b>CANCEL</b>	Only apply to reservation triggers
<b>CHANGE_PARAM</b>	
<b>JOB_PREEMPT</b>	This indicates that the trigger should preempt all jobs currently allocating resources assigned to the trigger's parent object. Only apply to reservation triggers.
<b>MAIL</b>	
<b>THRESHOLD</b>	
<b>INTERNAL</b>	

Value	Description
<b>EXEC</b>	

TriggerEventType

This enumeration specifies the event type of a trigger.

Value	Description
<b>CANCEL</b>	
<b>CHECKPOINT</b>	
<b>CREATE</b>	
<b>END</b>	
<b>EPOCH</b>	
<b>FAIL</b>	
<b>HOLD</b>	
<b>MIGRATE</b>	
<b>MODIFY</b>	
<b>PREEMPT</b>	
<b>STANDING</b>	
<b>START</b>	
<b>THRESHOLD</b>	

TriggerFlag

This enumeration specifies a flag belonging to a trigger.

Value	Description
<b>ATTACH_ERROR</b>	If the trigger outputs anything to stderr, Moab will attach this as a message to the trigger object.
<b>CLEANUP</b>	If the trigger is still running when the parent object completes or is canceled, the trigger will be killed.
<b>CHECKPOINT</b>	Moab should always checkpoint this trigger. See Checkpointing a Trigger in the Moab Workload Manager documentation for more information.
<b>GLOBAL_VARS</b>	The trigger will look in the name space of all nodes with the globalvars flag in addition to its own name space. A specific node to search can be specified using the following format: globalvars+node_id
<b>INTERVAL</b>	Trigger is periodic.
<b>MULTIFIRE</b>	Trigger can fire multiple times.
<b>OBJECT_XML_STDIN</b>	Trigger passes its parent's object XML information into the trigger's stdin. This only works for exec triggers with reservation type parents.
<b>USER</b>	The trigger will execute under the user ID of the object's owner. If the parent object is sched, the user to run under may be explicitly specified using the format user+<username>, for example flags=user+john:
<b>GLOBAL_TRIGGER</b>	The trigger will be (or was) inserted into the global trigger list.
<b>ASYNCHRONOUS</b>	An asynchronous trigger.
<b>LEAVE_FILES</b>	Do not remove stderr and stdout files.
<b>PROBE</b>	The trigger's stdout will be monitored.
<b>PROBE_ALL</b>	The trigger's stdout will be monitored.
<b>GENERIC_SYSTEM_JOB</b>	The trigger belongs to a generic system job (for checkpointing).

Value	Description
<b>REMOVE_STD_FILES</b>	The trigger will delete stdout/stderr files after it has been reset.
<b>RESET_ON_MODIFY</b>	The trigger resets if the object it is attached to is modified, even if multifire is not set.
<b>SOFT_KILL</b>	By default, a SIGKILL (kill -9) signal is sent to the kill the script when a trigger times out. This flag will instead send a SIGTERM (kill -15) signal to kill the script. The SIGTERM signal will allow the script to trap the signal so that the script can clean up any residual information on the system (instead of just dying, as with the SIGKILL signal).  NOTE: A timed-out trigger will only receive one kill signal. This means that if you specify this flag, a timed-out trigger will only receive the SIGTERM signal, and never the SIGKILL signal.

TriggerPeriod

This enumeration specifies the period of a trigger.

Value	Description
<b>MINUTE</b>	
<b>HOUR</b>	
<b>DAY</b>	
<b>WEEK</b>	
<b>MONTH</b>	

Trigger

Field Name	Type	Description
<b>id</b>	<b>String</b>	Trigger id - internal ID used by moab to track triggers

Field Name	Type	Description
<b>action</b>	<b>String</b>	For exec atype triggers, signifies executable and arguments. For jobpreempt atype triggers, signifies PREEMPTPOLICY to apply to jobs that are running on allocated resources. For changeparam atype triggers, specifies the parameter to change and its new value (using the same syntax and behavior as the changeparam command).
<b>actionType</b>	<a href="#"><u>TriggerActionType</u></a>	
<b>blockTime</b>	<b>Date</b>	Time (in seconds) Moab will suspend normal operation to wait for trigger execution to finish. Use caution as Moab will completely stop normal operation until BlockTime expires.
<b>description</b>	<b>String</b>	
<b>eventType</b>	<a href="#"><u>TriggerEventType</u></a>	
<b>expireTime</b>	<b>Date</b>	Time at which trigger should be terminated if it has not already been activated.
<b>failOffset</b>	<b>Date</b>	Specifies the time (in seconds) that the threshold condition must exist before the trigger fires.
<b>flags</b>	<a href="#"><u>Set&lt;TriggerFlag&gt;</u></a>	
<b>interval</b>	<b>Boolean</b>	When used in conjunction with MultiFire and RearmTime trigger will fire at regular intervals. Can be used with <a href="#"><u>TriggerEventType.EPOCH</u></a> to create a Standing Trigger. Defaults to false
<b>maxRetry</b>	<b>Integer</b>	Specifies the number of times Action will be attempted before the trigger is designated a failure.
<b>multiFire</b>	<b>Boolean</b>	Specifies whether this trigger can fire multiple times. Defaults to false.
<b>name</b>	<b>String</b>	Trigger name - can be auto assigned by moab or requested. Alphanumeric up to 16 characters in length
<b>objectId</b>	<b>String</b>	The ID of the object which this is attached to.

Field Name	Type	Description
<b>objectType</b>	<b>String</b>	The type of object which this is attached to. Possible values: <ul style="list-style-type: none"> <li>• vm - Virtual Machine</li> </ul>
<b>offset</b>	<b>Date</b>	Relative time offset from event when trigger can fire.
<b>period</b>	<a href="#"><u>TriggerPeriod</u></a>	Can be used in conjunction with Offset to have a trigger fire at the beginning of the specified period. Can be used with EType epoch to create a standing trigger.
<b>rearmTime</b>	<b>Date</b>	Time between MultiFire triggers; rearm time is enforced from the trigger event time.
<b>requires</b>	<b>String</b>	Variables this trigger requires to be set or not set before it will fire. Preceding the string with an exclamation mark (!) indicates this variable must NOT be set. Used in conjunction with Sets to create trigger dependencies.
<b>sets</b>	<b>String</b>	Variable values this trigger sets upon success or failure. Preceding the string with an exclamation mark (!) indicates this variable is set upon trigger failure. Preceding the string with a caret (^) indicates this variable is to be exported to the parent object when the current object is destroyed through a completion event. Used in conjunction with Requires to create trigger dependencies.
<b>threshold</b>	<b>String</b>	Reservation usage threshold - When reservation usage drops below Threshold, trigger will fire. Threshold usage support is only enabled for reservations and applies to percent processor utilization. gmetric thresholds are supported with job, node, credential, and reservation triggers. See Threshold Triggers in the Moab Workload Manager documentation for more information.
<b>timeout</b>	<b>Date</b>	Time allotted to this trigger before it is marked as unsuccessful and its process (if any) killed.
<b>unsets</b>	<b>String</b>	Variable this trigger destroys upon success or failure.

## Related topics

- [Standing reservations on page 1661](#)

## Fields: User's Permissions

**i** See the associated [Permissions on page 1593](#) resource section for more information on how to use this resource and supported operations.

### Additional references

Type	Value	Additional information
<b>Permissions resource</b>	permissions/users	<a href="#">Permissions on page 1593</a>
<b>Hooks filename</b>	permissions.users.groovy	<a href="#">Pre and post-processing hooks on page 1434</a>
<b>Distinct query-supported</b>	Yes	<a href="#">Distinct on page 1526</a>

## API version 3

UserPermission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the cached user permission.
<b>name</b>	<b>String</b>	The unique name of the user.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	The list of permissions.

Permission

Represents a permission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this role.
<b>action</b>	<b>String</b>	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	A description of this permission.
<b>fieldPath</b>	<b>String</b>	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	A human readable label for this permission.
<b>resource</b>	<b>String</b>	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	The type of the permission. Only 'api' type permissions are enforced.

### PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

## API version 2

UserPermission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the cached user permission.
<b>name</b>	<b>String</b>	The unique name of the user.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	The list of permissions.

Permission

Represents a permission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this role.
<b>action</b>	<b>String</b>	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	A description of this permission.
<b>fieldPath</b>	<b>String</b>	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	A human readable label for this permission.
<b>resource</b>	<b>String</b>	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	The type of the permission. Only 'api' type permissions are enforced.

### PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

## API version 1

UserPermission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of the cached user permission.
<b>name</b>	<b>String</b>	The unique name of the user.
<b>permissions</b>	<a href="#">List&lt;Permission&gt;</a>	The list of permissions.

Permission

Represents a permission

Field Name	Type	Description
<b>id</b>	<b>String</b>	The unique ID of this role.
<b>action</b>	<b>String</b>	The action that can be performed on the resource.
<b>description</b>	<b>String</b>	A description of this permission.
<b>fieldPath</b>	<b>String</b>	Field level ACL control, if null or '*', all fields are accessible, otherwise requests must match dot delimited path. Currently only checked when doing writable actions. Example - attributes.*: create update
<b>label</b>	<b>String</b>	A human readable label for this permission.
<b>resource</b>	<b>String</b>	The resource the permission applies to.
<b>resourceFilter</b>	<b>Map&lt;String, Map&gt;</b>	A map used to limit which resource instances this permission applies to. If this is null then the permission will apply to all instances of the resource. For api permissions the filter uses mongo query syntax.
<b>scope</b>	<a href="#">PrivilegeScope</a>	Whether this permission applies to the principal's tenant-associated resources or globally
<b>type</b>	<b>String</b>	The type of the permission. Only 'api' type permissions are enforced.

### PrivilegeScope

Some permissions and roles ignore tenants and apply globally. Others apply only to the resources associated with the principal's tenants.

Value	Description
<b>GLOBAL</b>	Describes a role or permission that applies globally, irrespective of the principal's tenants. This scope can be applied to any role or permission.
<b>TENANT</b>	Describes a role or permission that applies only to the resources associated with the principal's tenants. This scope can be applied to any role, but only to those permissions associated with tenanted resources (e.g. nodes, services, etc.).
<b>NONE</b>	Scope doesn't apply to some permissions. As of right now, all non-domain permissions (e.g. those created by Viewpoint) don't need a scope. NONE should therefore be assigned to all non-domain permissions.

#### Related topics

- [Permissions on page 1593](#)

# Moab Cluster Manager

## MCM overview

Moab Cluster Manager (MCM) is a Java-based graphical interface for managing the Moab Workload Manager. It allows users to submit jobs, schedule reservations, view job statistics, etc. in an easy, user-friendly way. This guide contains the following sections to help you set up and use MCM.

- [Getting Started on page 2435](#) explains how to get started using the MCM by installing it, connecting it to a Moab Workload Manager, and describing its main window (see
- [Workload on page 2442](#) deals with the submitting and viewing of jobs, reservations, and triggers. These functions are used to get work done by the system.
- [Resources on page 2471](#) explains how the Resources category gives administrators the ability to view, modify, and set policies and attributes while effectively diagnosing various system resources.
- [Credentials on page 2486](#) explains how the Credentials allow an administrator to view all credentials in the system and their various roles. New credential profiles can be added as well as modified.
- [Policies on page 2508](#) offers information about how the Policies section is intended to give control over Moab's various policies.

- [Reporting on page 2524](#) explains how MCM offers a wide assortment of customizable statistics whether they be customized charts, graphs, matrix statistics, quick charts, or reports.
- [Diagnostics on page 2538](#) explains how Diagnostics are intended to give an administrator quick and easy system information for diagnosing potential problems.
- [Miscellaneous on page 2538](#) contains varied MCM information that does not fit in other categories.

## Getting Started

### Getting Started Overview

Moab Cluster Manager (MCM) is a Java-based graphical interface for managing the Moab Workload Manager. It allows administrators to schedule reservations, view job statistics, manage policies, etc., in an easy, user-friendly way.

This chapter explains how to get started using the Moab Cluster Manager by installing it, connecting it to a Moab Workload Manager, and describing its main window.

## Installation

### Unix-Based installation

You will need to have Java SE 6 or higher installed on your system to run MCM. Java SE 6 comes with a Java Runtime Environment (JRE) version 1.6. You must obtain a JRE from Oracle. MCM will not run correctly on JREs from other vendors, such as GCJ or OpenJDK. The Linux build of MCM comes bundled with a JRE. However, if you are using the generic Unix version of MCM, you will need to download a JRE separately.

To see which version of Java you are running, you can type `java` at a command prompt.

```
$ java -version
java version "1.6.0_31"
Java(TM) SE Runtime Environment (build 1.6.0_31-b04)
Java HotSpot(TM) 64-Bit Server VM (build 20.6-b01, mixed mode)
```

These install instructions assume a basic familiarity with Unix/Linux file systems, and commands such as `ls`, `tar`, `mv`, etc.

The installation steps are as follows:

1. Download the latest version of the tar file from the Adaptive Computing web site:
  - a. The `mcm-x-[build number].linux.tar.gz` builds come with a bundled JRE.
  - b. The `mcm-x-[build number].tar.gz` builds do NOT include a JRE.
2. Move the tar file to your home directory, or another directory you have access to (i.e. `/home/username/`). Unpack the tar file (`tar -xzvf xxxx.tar.gz`).

3. Change directories to the newly unpacked MCM directory.
4. Start MCM by running the MCM script as shown below:

```
$ mv mcm-8.0.0.x-xxxx.tar.gz /home/bob
$ cd /home/bob
$ tar zxvf mcm-8.0.0.x-xxxx.tar.gz
$ cd mcm-8.0.0.x-xxxx/
$ ./mcm &
```

The MCM script will check for the existence of a compatible JRE and then run MCM. After accepting the license you will see the Connection Wizard. To see how to configure MCM to connect to Moab Workload Manager, see the Connection Wizard section.

## Windows installation

1. Download the installation executable from the Adaptive Computing web site. The `mcm-8.0.0.x-[build number].exe` build contains a Windows installer that will set up MCM on your system. This build comes bundled with a compatible Java Runtime Environment (JRE).
2. Double-click the installation file. The MCM installer will guide you through the installation process.

**i** The default target folder is either `C:\Program Files\Moab Cluster Manager` or `C:\Program Files (x86)\Moab Cluster Manager` depending on what version of Windows you are running.

3. The installer will create Start Menu and Desktop icons that can be used to run MCM.
4. Double-click the Moab Cluster Manager icon on the Desktop to run MCM. After accepting the license you will see the Connection Wizard. To see how to configure MCM to connect to Moab Workload Manager, see the [Connection Wizard](#) section.

**i** If you're running Windows 7, MCM by default will NOT have permissions to create log files. This is because MCM will not have permission to write to subfolders of `C:\Program Files` or `C:\Program Files (x86)` unless it is run as an administrator. This may not be a concern. However, if you need to see log files, you can change the log file path to a folder where the user running MCM has permission to write to. To do this, edit the `C:\Program Files (x86)\Moab Cluster Manager\conf\log4j.properties` file and change the **log4j.appender.ROLL.File** property. For example, to cause logs to be written to `C:\Users\Bob\AppData\Local`, where Bob is the user running MCM, modify the property as shown below:

```
log4j.appender.Roll.File=${user.home}/AppData/Local/mcm.log
```

## Connection wizard

The Connection Wizard provides three connection options for the Moab Cluster Manager: **Remote Connection**, **Local Connection**, and **Online Demonstration**.

Connection	Description
<b>Remote Connection</b>	Connects to a remote Moab Workload Manager server over SSH.
<b>Local Connection</b>	Connects to a locally hosted Moab Workload Manager server.
<b>Online Demonstration</b>	Automatically logs in to an online demonstration cluster for a preview of the product.

## Remote Connection

The Remote Connection feature allows you to securely connect to a remote Moab Workload Manager server. Here is a description of each of the connection options:

### Host Name & Port

#### **Host Name (or IP Address)**

The Host Name (or IP address) of the server that is running Moab Workload Manager. If you do not know the host name or IP address of the server, please consult your system administrator.

#### **Port**

The port on which SSH is running on the remote server (the default is 22). If you do not know which port to use, please consult your system administrator.

### Authentication Options

#### **Password Authentication**

This option tells Moab Cluster Manager to authenticate by prompting the user for a password. Consult with your system administrator for information regarding your user name, password and the type of authentication used.

#### **Keyboard Interactive Authentication**

This option tells Moab Cluster Manager to interactively prompt for authentication information.

#### **SSH Key Authentication**

This option tells Moab Cluster Manager to connect to the remote computer using only the user name and a private key file. Consult your system administrator for information regarding your user name, private key and the type of authentication used.

 Keys should only be generated with `ssh-keygen`.

### Ask for SSH Key Passphrase

Some private keys require a passphrase to be entered before it will allow a user to authenticate. In this case this box should be checked, otherwise an empty passphrase will be used for authentication.

### Connection Settings

#### User Name

This is the name used to login to the remote computer. Because MCM is an administrator tool, the user must be configured as an [ADMINCFG\[1\]](#) in `moab.cfg`. Consult your system administrator for information regarding your user name or password.

#### Path to Moab Workload Manager Client Commands (i.e. showq)

The directory containing the Moab Client Commands, such as `showq`, `mschedctl`, `mdiag`, etc. This is not the location of the Workload Manager, but instead the location of the commands that control the Workload Manager. This location is usually `/opt/moab/bin`.

#### Private Key Path

If SSH key Authentication is being used, this field is for the path of the private key file.

### Load, save or delete stored sessions

#### Auto Connect On Next Session

This option sets the Moab Cluster Manager to automatically connect to the specified saved session the next time it is run.

#### Saved Sessions

This field is where a user is able to save his/her remote connection settings so that they don't need to be entered each time.

#### Load Button

This button will load the selected saved session in the list to the left.

#### Save Button

This button will save a session according to the name typed in the **Saved Sessions** field.

#### Delete Button

This button will delete the saved session that is selected in the list to the left.

#### Connect Button

This button will open a connection to a remote Moab Workload Manager. It will attempt to authenticate the user based on the given settings, and then open the Moab Cluster Manager.

#### Quit Button

Clicking this button will close the Moab Cluster Manager Connection Wizard.

## Local Connection

This option connects to a Moab Workload Manager running on the local machine. The only option for this mode is the Path for the Moab Workload Manager Client Commands.

### Path to Moab Workload Manager Client Commands (i.e. showq)

The directory containing the Moab Client Commands, such as `showq`, `mschedctl`, `mdiag`, etc. This is not the location of the Moab Workload Manager, but instead the location of the commands that interact with the Moab Workload Manager. This location is usually `/opt/moab/bin`.

### Connect Button

This button will open a connection to a local Moab Workload Manager and then open the Moab Cluster Manager.

### Quit Button

Clicking this button will close the Moab Cluster Manager Connection Wizard.

## Online Demonstration

The Online Demonstration is a free, online demonstration cluster for users to preview. This connection option will automatically log in to the demonstration cluster.

### Connect Button

This button will connect Moab Cluster Manager to the Adaptive Computing demonstration Moab Workload Manager.

### Quit Button

Clicking this button will close the Moab Cluster Manager Connection Wizard.

## View summary

The Moab Cluster Manager main window provides an overview of the current state of the cluster. There are four parts of this main window: the Main Menu Bar, the Navigation pane, the Summary screen, and the System Utilization Bar.

## Main Menu Bar

The Main Menu Bar is located across the top of the Moab Cluster Manager window. Through this menu (Main, Configure, Create, etc.), all Moab Cluster Manager features can be accessed. The services unique to this menu are:

- Console window
- Save System Snapshot window
- Moab Cluster Manager Preferences window

- Plugin Manager window
- About window

## Navigation Pane

The Navigation pane is a directory of all the services that the Moab Cluster Manager can provide to users and administrators. The availability of some services depends on the user's privileges, as determined by the **ADMINCFG** level defined in the `moab.cfg` file. The chapters of this User Guide mimic the layout found in the Navigation pane.

## Summary Screen

The largest area in the Moab Cluster Manager main window is the Summary screen. This screen is intended to give general information about the system that Cluster Manager is currently connected to.

### Scheduler Information

This panel displays the following information about the scheduler:

#### **Name**

The name of the scheduler. Has no impact on operation.

#### **Host**

This refers to the host computer where the Resource Manager is running.

#### **Port**

The specific port that the scheduler is operating on.

#### **Mode**

The operating mode of the scheduler. Mode options are shown in this table:

Mode	Description
<b>NORMAL (default)</b>	Normal operation, controls the Resource Manager as configured
<b>MONITOR</b>	MONITOR mode behaves identical to NORMAL mode except the ability to start, cancel, or modify jobs is disabled. This allows safe diagnosis of the scheduling state and behavior using the various diagnostic client commands.
<b>INTERACTIVE</b>	Like NORMAL mode, except Moab sends the desired change request to the screen and asks for permission to complete it.

#### **Status**

Indicates whether the scheduler is running, down, or paused.

### Node Summary

This panel displays a high-level view of the state of the nodes found within the cluster. Click on any label to obtain a detailed list of nodes in the given category. Category descriptions:

#### **Busy Nodes**

Busy nodes include all nodes which are actively executing batch jobs. A node will be listed as busy even if it is only partially loaded with jobs.

#### **Idle Nodes**

Idle nodes include all nodes which are available but are currently not running any jobs.

#### **Down Nodes**

Down nodes include all nodes which have reported major software, hardware, or batch failures or have been marked 'down' or 'offline' by an administrator.

#### **Total Nodes**

The total nodes category includes all nodes in the cluster and is a sum of the busy, idle, and down nodes listed above.

### Job Summary

This panel displays a high-level view of the state of the jobs found within the cluster. Click on any label to obtain a detailed list of jobs in the given category. Category Descriptions:

#### **Running Jobs**

Running jobs include all jobs which are actively executing or performing post-execution clean-up. This includes jobs in the states starting, running, or exiting.

#### **Eligible Jobs**

Eligible jobs include all jobs which are in state idle and are not blocked by holds, usage limits, or other policies. Eligible jobs typically will run as soon as resources become available and may already have a reservation in place.

#### **Blocked Jobs**

Blocked jobs include all jobs which cannot run due to reasons other than resource availability. Jobs may be blocked by job holds, resource manager level policies, scheduler job usage policies, job deadline constraints, or other factors. Clicking on the label will bring up the detailed blocked job list which will include additional information in the blocked reason column.

#### **Total Jobs**

The total jobs category includes all jobs in the cluster and is a sum of the running, eligible, and blocked jobs listed above.

### User Information

This panel displays the user information of whoever started the Moab Cluster Manager.

#### **User**

The name of the user running Moab Cluster Manager.

**Group**

The name of the user's group.

**Account**

Any accounts the user may belong to.

**Class**

Classes the user has access to.

**QoS**

Any QoS accounts the user may belong to.

[User Job Summary](#)

Displays information concerning jobs run by the current user.

**Running Jobs**

Jobs run by the current user.

**Eligible Jobs**

Users jobs that are waiting in the queue.

**Blocked Jobs**

Users jobs that have been blocked, either by policy or user.

**Total Jobs**

Total jobs from the user.

## System Utilization Bar

The System Utilization bar appears at the bottom of the screen. It displays historical system utilization as it pertains to utilized processors as captured by Moab profiling intervals. Also note that processors utilized are only measured once a job is finished.

# Workload

## Workload Overview

The workload category of features deals with the submitting and viewing of jobs, reservations, and triggers. These functions are used to get work done by the system.

# Jobs

## Create Job

A cluster runs programs. A job tells a cluster when, where, and how to run the programs. The **Submit Job** window, often referred to as a job submission window, is how a user creates a job.

You can choose between a basic options view, which displays the options described in the [Basic](#) table below, and an advanced options view, which displays the options described in all the tables below. To switch views, click **View** in the main menu bar and check or uncheck the **Show Advanced Options** checkbox.

### Job Creation

#### [Basic](#)

##### Template

If there are job templates that are selectable, the user can do so here. Any attributes associated with the job templates will be mapped onto the submitted job.

##### Script File

This field allows an administrator to search for a script to run.

##### Nodes

This allows an administrator to select where the job will run from the following options:

Option	Description
<b>Number of Nodes</b>	A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node. If this field is set to 0, Workload Manager assigns the job to 1 node unless the <b>Node List</b> field is populated.
<b>Node List</b>	A node is a computer consisting of 1 or more processors. This field allows the user to define which nodes a job requires to execute. If a node list is not specified the nodes needed for the job are gathered from the nodes field.
<b>VM List</b>	The specific virtual machines the job will use.

#### [Job Information](#)

##### Job Name

A user can attach a custom name to a job to assist him/her in identifying the job. The name is provided only for the user's convenience and does not affect any policies or settings.

**User Job Priority**

The higher a job's priority, the sooner it will start. By changing this field, a user can reduce the job's priority and change the order in which the job starts. This field is usually utilized to execute a user's jobs in a specific order. This field only supports negative numbers with the exception of 0. A 0 User Job Priority will not delay the job from starting. However, the higher a negative number, the lower a job's priority. For example, a job with a User Job Priority of -100 will allow more jobs to start before it starts, thus postponing the job from starting.

Resources**Processors per Node**

All jobs require at least 1 processor. If this field is not used, the processors are calculated using the available processors on a node. If a processor and a node are not requested, Workload Manager assigns 1 node to the job.

**Memory per Node**

Some jobs require a specific amount of memory. This field allows a job to request the memory it needs for each node. It should be noted that this field is not the total memory across the entire cluster but only the memory on each node needed by the job. Workload Manager will start this job only on the nodes that have sufficient memory. If this field is not used, Workload Manager will start the job on any available node.

**Swap per Node**

Some jobs require a specific amount of swap. This field allows a job to request the swap it needs for each node. It should be noted that this field is not the total swap across the entire cluster, but the swap on each node needed by the job. Workload Manager will start this job only on the nodes that have sufficient swap. If this field is not used, Workload Manager will start the job on any available node.

**Operating System**

If an operating system is selected, Moab will try to run the job on any nodes with the specified operating system.

**Architecture**

If an architecture is selected, Moab will try to run the job on any nodes with the specified architecture.

**Required Node Features**

Some jobs require a specific feature on a node. A feature is a custom tag attached to a specific list of nodes. Consult your system administrator for specific information regarding each tag.

**Partition**

Clusters are often divided into different sections. These sections are commonly called partitions. Users can only request one specific partition for their job. Consult your system administrator to learn which partition is the best suited for your job.

**Reservation**

If a reservation is specified, the job will execute only on the nodes of the selected reservation.

## Generic Resources

### **Type**

The name of the preconfigured generic resource that you want to request for this job.

### **Quantity**

The number of this type of generic resource that you want to request for this job.

## Data Management

### **Execution Directory**

Some jobs need to be executed in a specific location on each node. This field allows a user the ability to define that location. By default, the job is executed in the user's home directory. Consult your system administrator for information regarding your home directory.

### **Output File**

This field gives the user the ability to customize the location of where the standard output stream will be written. The standard output stream of the job will be placed in a directory relative to the execution directory, as shown in the following example:

```
> test12/stdout.txt
```

### **Error File**

This field gives the user the ability to customize the location of where the standard error stream will be written. The standard error stream of the job will be placed in a directory relative to the execution directory, as shown in the following example:

```
> test12/stderr.txt  
> test12/stderr.txt
```

## Credential Information

### **User**

This field defines the name of the user under whom the job will execute. Only users with Admin1 rights can change this field. See [User Access](#) for more information.

### **Group**

This field defines the name of the group under which the job will execute. Only the groups available to the user are displayed.

### **Account**

This field defines the name of the account under which the job will execute. Only the accounts available to the user are displayed.

### **Class**

This field defines the name of the class under which this job will execute. Only the classes available to the user are displayed.

**Quality of Service (QoS)**

This field defines the name of the quality of service (QoS) under which this job will execute. Only the QoSs available to the user are displayed.

Job Dependencies

**Job Dependencies**

This field allows an administrator to specify which other jobs this job is dependent on. The dependency types are as follows:

Dependency Type	Description
<b>After</b>	The job starts any time after specified jobs start executing.
<b>Afterok</b>	The job may start any time after all specified jobs have completed successfully.
<b>Afternotok</b>	The job may start any time after all specified jobs have completed unsuccessfully.
<b>Afterany</b>	The job may start at any time after all specified jobs have completed regardless of status.

Flags/Options

**Preemptible**

A job that is preemptible can be suspended or re-queued by higher priority jobs.

**Restartable**

If a job experiences a failure during execution, the user must resubmit the job to Workload Manager. However, a job that is restartable is automatically restarted by Workload Manager in the event of a failure.

**Preemptor**

The job may preempt other jobs which have the *PREEMPTEE* flag.

**Hold**

A hold can only be placed upon jobs that have not yet executed. A hold stops or halts a job from running until the user or an administrator releases the hold.

Time Frame

**Start Time**

Some jobs require a specific amount of time before they are allowed to start. This field allows the user the ability to define the earliest time that the job can begin. By default, a job may start as soon as resources become available.

### Estimated Start Time Calculator

This button displays Moab's text-based output that determines when a job can start.

### Estimated Start Time Table

This button displays a table of the estimated start times for jobs of different processor sizes.

### [Email Notification](#)

#### On Job Completion

When a job finishes execution an email notification will be sent to the user stating this.

#### On Job Execution

When a job begins execution an email notification will be sent to the user stating this.

#### On Job Failure

When a job cannot start or crashes during execution an email notification will be sent to the user stating this.

 User email addresses may be specified in the [Create/Manage Users](#) page.

## Modify a Job

### Summary

A cluster runs programs. A job tells a cluster when, where, and how to run the programs. The Modify Job window allows a user to modify an existing job. The fields that can be modified are user job priority, system priority, and duration. If multiple jobs are selected, then QoS can also be modified.

### [Overview](#)

#### ID

This field displays the unique ID given to the job by Workload Manager.

#### Name

Users can attach a custom name to the job to allow them to easily identify their jobs. The name does not change any Workload Manager settings or prioritizations. If a name has been attached, it will appear in this field.

#### State

This field displays the execution status of the job. For example, running, stopped, executing, idle, blocked, etc.

### [Credentials](#)

#### User

This field displays the user ID under which the job executes.

**Group**

This field displays the group ID under which the job executes.

**Account**

This field displays the account ID used by the job.

**Class**

This field displays the class/queue ID used by the job.

**Quality of Service (QoS)**

This field displays the quality of service (QoS) ID used by the job.

[Resources](#)**Allocated Node List**

A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node. The node list is a list of the nodes that the job is using.

**Allocated Partition**

A partition is a logical construct that divides available resources. Any single resource (compute node) may only belong to a single partition. This field displays the name of the partition to which the job is assigned.

**Processor per Task**

A task is a group of resources that must all be on the same node. One resource in that group is a processor. This field displays the number of processors in each task that the user's job requires.

**Required Minimum Nodes**

A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node. Some jobs are large and require a larger number of nodes to run. This field displays the minimum number of nodes the job requires to run.

**Required Minimum Tasks**

A task is a group of resources that must all be on the same node. This field displays the minimum number of tasks, or groups of resources, the user's job requires on each node.

**Tasks per Node**

A task is a group of resources that must all be on the same node. This field displays the number of tasks, or groups of resources, that the job requires on each node.

[Job Attributes](#)**PAL (Partition Access List)**

This field displays the partitions available for this user. Clusters can be divided into different sections commonly called partitions. Consult your system administrator to learn which partition is the best suited for your job.

### **Start Count**

This field displays the number of times the job has started. Multiple job starts could indicate job failure or rejection.

### **Start Priority**

This field displays the start priority for the job. With some exceptions, the higher a job's priority over other jobs, the sooner it will begin to execute.

### **User Job Priority**

With a few exceptions, a job with a high priority will begin sooner depending on how much greater its priority is to that of other jobs. A user has the ability to reduce the job's priority and, in effect, delay the job's start time by changing this field. This option is usually utilized by users who desire their jobs to execute in a specific order. This field only supports negative numbers with the exception of 0. A 0 user job priority will not delay the job from starting. However, a job's priority will decrease as the priority number decreases. For example, a user job delay priority of -100 will allow more jobs to start before it than a job with a user job priority of -10. It should be noted that the user job priority literally lowers the start priority of a job.

### **System Priority**

With some exceptions, the higher a job's priority over other jobs, the sooner it will begin to execute. This field allows an administrator the ability to start jobs immediately. This field adds 1,000,000,000 and the administrator priority to the start priority of the job.

## Time Frame

### **Start Time**

This field displays the date and time that the job started or is scheduled to start.

### **Duration**

This field displays the job's estimated execution time. If a job requires more time than the specified duration, duration violation policies come into effect. Consult your system administrator for more information regarding these policies. If no duration is specified, a default walltime is applied. Consult your system administrator for more information regarding your cluster's default walltime.

### **Completed Duration**

This field displays the amount of time the job has been running, or the time that has passed between the **Start Time** and now.

### **Queue Time**

This field displays the amount of time the job has been waiting to start.

## Generic Resources

### **Type**

The name of the preconfigured generic resource that you want to request for this job.

### **Quantity**

The number of this type of generic resource that you want to request for this job.

## List Jobs/Job Templates

You can change the set of columns MCM displays in the Jobs table by clicking **View > Change Column Group** in the main menu bar and selecting one of the following options.

Option	Columns Displayed
<b>Summary</b>	Job ID, Job Name, State, User, Start Time, Used Wallclock, Wall Clock, Allotted Nodes, Required Nodes
<b>Credentials</b>	Job ID, Queue Status, State, User, Group, Class, Account, QoS
<b>Time</b>	Job ID, State, Start Time, Used Wallclock, Completion Time, Submission Time, System Start Time, Wall Clock, Earliest Start Time, Latest Completion Time
<b>Node Information</b>	Job ID, State, Allocated Node List, Master Node, Node List
<b>IDs</b>	Job ID, State, Global Job ID, System ID
<b>Required Resources</b>	Job ID, State, Allocated Node List, Partition, Allocated Nodes, Required Procs, Req. Node Memory
<b>Utilized Resources</b>	Job ID, State, Mem. Seconds Utilized, Proc Seconds Dedicated, ProcSeconds Utilized
<b>Reservation</b>	Job ID, State, Reservation
<b>Script</b>	Job ID, State, Script File, Arguments, Initial Working Directory, Variables
<b>Priority</b>	Job ID, State, Run Priority, System Priority, User Job Priority, Start Priority
<b>Diagnostic</b>	Job ID, State, Suspend Duration, Execution Eligibility Time, Hold, Block Reason, Expected State, Bypass
<b>Comments</b>	Job ID, State, Messages

## List Job Fields

When you click **View > Visible Columns...** in the main menu bar, you can manually add or remove individual columns in your **Jobs** table. The following table describes each column:

Field	Field Information
<b>Account</b>	This field displays the account ID used by the job.
<b>Allocated Node List</b>	A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node. The allocated node list is a list of the nodes that the job is using.
<b>Allocated Nodes</b>	This field displays the number of nodes allocated to the job.
<b>Arguments</b>	Some programs provide users with options. This field allows the user the ability to view those options. A user should consult the program documentation to learn about the available options.
<b>Block Reason</b>	This field displays diagnostic messages related to the job.
<b>Bypass</b>	This field displays the number of times another job of a lower priority started before this job.
<b>Class</b>	This field displays the class id used by the job.
<b>Completion Time</b>	This field displays the date and time the job finished execution.
<b>Earliest Start Time</b>	This field displays the user-specified date and time the job is available to start. Workload Manager will not start the job until after this specified date and time.
<b>Execution Eligibility Time</b>	This field displays the time the job was eligible for execution. The format is <code>hours:minutes:seconds</code> .
<b>Expected State</b>	This field displays the estimated execution status of the job. For example, running, stopped, executing, idle, blocked, etc.
<b>Flags</b>	Cluster Manager schedules jobs differently according to their flags. Possible flags are hold, interactive, restartable, and preemptible. Refer to the <a href="#">create job documentation</a> for definitions of the flags.
<b>Generic Attributes</b>	This field displays a custom attribute attached to the job.
<b>Global Job ID</b>	The global job ID is used when multiple resource managers are being used.

Field	Field Information
<b>Group</b>	This field displays the group id under which the job executes.
<b>Hold</b>	This field displays whether the job has a hold. A hold can only be placed on jobs that have not started. A hold stops or halts a job from running until the user or an administrator releases the hold.
<b>Initial Working Directory</b>	Some jobs must execute in a specific location on each node. This field allows a user the ability to define that location. By default, the job is executed in the user's home directory. Consult your system administrator for information regarding your home directory.
<b>Is Template Selectable?</b>	This field displays whether the job was created from a selectable template.
<b>Is Template?</b>	This field displays whether the job was created from a template.
<b>Job ID</b>	This field displays the job's unique ID given by Workload Manager.
<b>Job Name</b>	A user can attach a custom name to the job to allow him/her to easily identify their job. The name does not change any Workload Manager settings or prioritizations. If a name has been attached it will appear in this field.
<b>Latest Completion Time</b>	This field displays the date and time the job must finish execution.
<b>Master Node</b>	In a cluster, one specific node is in charge of communication with all the other nodes on the cluster. This node is often referred to as the master node or the head node. This field will display the name of the master node.
<b>Mem. Seconds Utilized</b>	Memory seconds utilized is defined as the total amount of memory used by the job times the number of seconds the memory was used. The value is calculated as a sum total of all the memory on the cluster and not on a per node basis.
<b>Messages</b>	This field allows the user to add a comment to the job.
<b>Node Features</b>	This field displays whether the job prefers a node with a specified feature.
<b>Node List</b>	This field displays the list of nodes that the job requires to execute.

Field	Field Information
<b>Partition</b>	Clusters are often divided into different sections. These sections are commonly called partitions. Users can only request one specific partition for the job. Consult your system administrator to learn which partition is the best suited for your job.
<b>Proc Seconds Dedicated</b>	Processor seconds dedicated is the total number of processors reserved by Workload Manager for the job times the amount of time, in seconds, that the processors were reserved. The value is calculated as a sum total of all the processors on the cluster and not on a per node basis.
<b>Proc Seconds Utilized</b>	Processor seconds utilized is the total number of processors used by the job times the number of seconds the processors were reserved. The value is calculated as a sum total of all the processors on the cluster and not on a per node basis.
<b>QoS (Quality of Service)</b>	This field will display the quality of service (QoS) id used by the job.
<b>QoS (Quality of Service) Required</b>	This field displays the required QoS for this job.
<b>Queue Status</b>	This field displays the job's status in the job queue. It indicates whether the job is currently Active, Eligible, Blocked, etc.
<b>Req. Access</b>	This field displays the required access for the job.
<b>Req. Allocated Node List</b>	A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node. The allocated node list is a list of the nodes that the job is using.
<b>Req. Allocated Partition</b>	Clusters are often divided into different sections. These sections are called partitions. Users can only request one specific partition for each job. Consult your system administrator to learn which partition is the best suited for your job.
<b>Req. Architecture</b>	Some jobs require a specific node architecture. This field allows a user to view the architecture required by this job.
<b>Req. Class</b>	This field displays the required class for this job.
<b>Req. Disk per Task</b>	A task is a group of resources that must all be on the same node. One resource in that group is disk space. This field displays the amount of disk in each task that the job requires.

Field	Field Information
<b>Req. Memory Per Task</b>	A task is a group of resources that must all be on the same node. One resource in that group is memory. This field displays the amount of memory in each task that the job requires.
<b>Req. Min Nodes</b>	A node is a computer consisting of 1 or more processors. This field displays the minimum number of nodes required for the job to execute.
<b>Req. Network</b>	This field displays the network required by this job.
<b>Req. Node Disk</b>	This field displays the required amount of disk space the job needs on each node. This field is not the total disk across the entire cluster but only the disk space on each node.
<b>Req. Node Memory</b>	This field displays the requested amount of memory it needs for each node. This field is not the total memory across the entire cluster but only the memory on each node.
<b>Req. Node Proc</b>	This field displays the processors required by this job.
<b>Req. Node Swap</b>	This field displays the required swap space the job needs for each node. It should be noted that this field is not the total swap across the entire cluster but only the swap on each node.
<b>Req. OS</b>	This field allows a user to view the operating system required by this job.
<b>Req. Partition</b>	Clusters are often divided into different sections. These sections are called partitions. Users can only request one specific partition for each job. Consult your system administrator to learn which partition is the best suited for your job.
<b>Req. Proc Per Task</b>	A task is a group of resources that must all be on the same node. One resource in that group is a processor. This field displays the number of processors in each task that the job requires.
<b>Req. Set</b>	This field displays the groupings of nodes this job requires.
<b>Required Swap Per Task</b>	A task is a group of resources that must all be on the same node. One resource in that group is swap space. This field displays the amount of swap in each task that the job requires.
<b>Req. Tasks Per Nodes</b>	A task is a group of resources that must all be on the same node. This field displays the number of tasks that the job requires on each node.

Field	Field Information
<b>Required Memory</b>	This field allows a job to request the memory it needs for each node. This field is not the total memory across the entire cluster but only the memory on each node needed by the job. Workload Manager will start this job only on the nodes that have sufficient memory.
<b>Required Nodes</b>	A node is a computer consisting of 1 or more processors. A job requires at least 1 processor to execute and therefore must use at least 1 node.
<b>Required Procs</b>	This field displays the number of processors used by this job.
<b>Reservation</b>	A user can specify a reservation for this job. If a reservation is specified the job will execute only on that reservation's nodes.
<b>Run Priority</b>	This field is used by jobs that are preemptible to decide which job should be preempted. With a few exceptions, the higher a job's priority, the more likely it will be preempted.
<b>SLA (Service-Level Agreement) Violation</b>	This field indicates the job service-level agreement violation, if any exist.
<b>Script File</b>	This field displays which script file the job executes.
<b>Start Count</b>	This field displays the number of times the job has attempted to start executing.
<b>Start Priority</b>	This field displays the start priority of the job. With a few exceptions, a job with a high priority will begin sooner depending on how much greater its priority is to other jobs.
<b>Start Time</b>	This field displays the date and time the job started.
<b>State</b>	This field displays the execution status of the job. For example, running, stopped, executing, idle, blocked, etc.
<b>Submission Time</b>	This field displays the time the job was first created. The format is hours:minutes:seconds. If the exact date is desired moving the mouse over the value will display the exact date of the submission time.
<b>Suspend Duration</b>	This field displays the time the job was in a suspended state. The format is hours:minutes:seconds.
<b>System ID</b>	The system job ID is used when multiple resource managers are being used.

Field	Field Information
<b>System Priority</b>	With a few exceptions, a job with a high priority will begin sooner depending on how much greater its priority is to other jobs. This field allows an administrator the ability to start jobs immediately. This field adds 1,000,000,000 and the administrator priority to the job's priority.
<b>System Start Time</b>	This field displays the time when the job started.
<b>Used Wallclock</b>	This field displays the actual execution time of the job. The format is hours:minutes:seconds. The white space indicates the remaining time left before the job reaches its requested Wallclock time. The colored section indicates the amount of Wallclock that has been used. A red bar indicates that the job has violated its Wallclock limit. Refer to the Wallclock field for the job Wallclock.
<b>User</b>	This field will display the user id under which the job is executing.
<b>User Job Priority</b>	With a few exceptions, a job with a high priority will begin sooner depending on how much greater its priority is to other jobs. A user has the ability to reduce the job's priority and, in effect, delay its start time by changing this field. This option is usually utilized by users who desire their jobs to execute in a specific order. This field only supports negative numbers with the exception of 0. A 0 user job priority will not delay the job from starting. However, a job's priority will decrease as the priority number decreases. For example, a user job delay priority of -100 will allow more jobs to start before it starts than a user job priority of -10.
<b>Variables</b>	This field displays the variable(s) that are set on the job.
<b>Wallclock</b>	This field displays the estimated amount of time that the job will execute. If a job requires more time than the specified duration, duration violation policies come into effect. Consult your system administrator for more information regarding these policies. If no duration is specified, a default walltime is applied. Consult your system administrator for more information regarding your cluster's default wall time.

## Job Timeline

The Job Timeline window displays each executing job in a diagram. On the left side of the Now line is the amount of a job that has completed. The right side is the remaining execution time.

## Job Outlines

Job outlines are settings saved from the create/submit job window. This window, accessed through the Submit Job page by clicking **Actions > Open Job Outline** or **Actions > Save Job Outline** in the main menu bar, allows job outlines to be saved or opened either locally on the machine that Moab Cluster Manager is running on, or remotely on the machine that Moab Workload Manager is running on.

## Outline Information

This section displays information of the currently loaded job outline.

### Outline Name

This field allows a user to change the saved file name of the outline.

### Job Name

This field displays the custom name for the job outline.

### Script Name

This field displays the script that is used in the outline.

## Outline Window

- Local Job Outlines — Outlines found on the machine running Moab Cluster Manager.
- Outline Directories — Directories where outlines can be found.

### Outline Operations

#### Open

This button retrieves the selected outline information and places it in the outline information fields.

#### Save

This button saves the selected outline.

## Job Workflows

A job workflow is a visual representation of the dependency relationships between jobs. The workflow shows which jobs depend on others. Since a job can depend on any number of jobs and vice versa, the workflow can show the whole dependency relationship for any given job.

To view a job workflow, select a job from the input options at the bottom of the page. The system then determines all jobs that depend or are dependent on this target job. The results are displayed graphically in a timeline window of the main page. Jobs are placed on the graph according to their start and end times.

On the graph, time spans from left (earliest time) to right (latest time). The time window is dependent on the workflow, as the earliest time shown is the earliest job's start time and the end time is its latest job's end time. If the current time is included in the time window, a vertical bar on the graph displays the current time. Keep in mind however that some workflows are entirely in the future and others are only in the past, in which case no vertical bar will be shown.

A dependency is represented as an arrow on the graph. Jobs that are dependent on others will have an arrow pointing to it representing the expected flow of the jobs, or in other words the order in which the jobs will run. There are various dependency types and therefore various job states that must be represented. Double clicking on any job will bring up more details about the job in the modify job window.

# Reservations

## Create Reservation

### Summary

A reservation sets apart resources during a particular time frame for a particular owner. Reservations usually reserve resources such as nodes or processors on the cluster. The Create Reservation window allows you to define what resources a reservation requires, as well as the time frame for the reservation.

In addition to being able to create a basic reservation, this window also allows you to create a recurring reservation. A recurring reservation, also referred to as a standing reservation or a reservation generator/creator, provides the user with the option of having reservations automatically created according to a desired time frame. For example, if a user wants to create a reservation every Tuesday and Thursday starting at 11 am and ending at 4 pm, a recurring reservation would fulfill this need. It should be noted that a recurring reservation could potentially not be able to create a reservation if the resources are already dedicated to another reservation or job. To reduce the possibility of this occurring, the day/week depth field can be increased so Workload Manager will create the reservations further in advance.

### *Panels Containing Required Parameters*

#### Reservation Information

This panel allows the user to specify the reservation name and owner.

#### **Reservation Name**

This field allows a user to create a name for the reservation. Workload Manager appends a numerical value to the end of the reservation name which allows users the ability to enter duplicate reservation names without affecting a previous reservation.

#### **Owner**

An owner is a user, group, account, class, or quality of service. A reservation can reserve only the resources that the owner can access. This field allows a user to select the owner of the reservation.

#### Access Control List

This panel allows the user to specify what credentials have permission to access the reservation. At least one credential is required to be in the Access Control List for a reservation, otherwise it would not be very useful. The user may select from 5 different types of credentials - users, groups, accounts, classes, and quality-of-services. Any of the credentials in the Access Control List have permission to use the reservation.

#### **Add**

Pops up a window that allows the user to select credentials of a specific type to add to the Access Control List

**Clear**

Removes all credentials from the Access Control List

Resources

Allows the user to specify what resources will be.

**Host List/Host Expression**

Users often require specific nodes for their applications. A list of nodes required by the user is called a host list. If a host list is not specified, the task count must be specified.

**Task Count**

A task is a group of resources that must all be on the same node. This field defines how many groups of resources will be required to create this reservation. If the task count is not specified the host list must be specified.

**Define Tasks**

A task is a group of resources that must all be on the same node. This button displays a window allowing a user to define what those resources will be in a task.

Field	Description
<b>Memory Per Task</b>	This field provides the user with the option of requesting the amount of memory, in megabytes, a job needs on a node.
<b>Swap Per Task</b>	This field provides the user with the option of requesting the amount of swap, in megabytes, a job needs on a node.
<b>Disk Per Task</b>	This field provides the user with the option of requesting the amount of disk space, in megabytes, a job needs on a node.
<b>Procs Per Task</b>	This field provides the user with the option of requesting the number of processors a job needs on a node.

Reservation Time Frame Tabs:

**Once**

Basic reservation will be created for the desired start and end time.

Field	Description
<b>Start Time</b>	This field allows the user to select the day and time when the reservation begins.
<b>End Time</b>	This field allows the user to select the day and time when the reservation ends.

**Daily**

A recurring reservation is initialized to automatically create a reservation on the desired days at the desired start and end time. The recurring reservation continually generates new reservations. The **day depth** is used to decide when a reservation is created. For example, if a reservation starts 4 days from now and the **day depth** is set to 2 days the reservation will not be created for 2 more days.

Field	Description
<b>Days</b>	This field allows the user to select the days of the week a reservation is created.
<b>Start Time</b>	This field allows the user to select the time of day the reservation begins.
<b>End Time</b>	This field allows the user to select the time of day the reservation ends.
<b>Day Depth</b>	This field allows the user to specify how many days in advance Workload Manager should create recurring reservations.

**Weekly**

A recurring reservation will be initialized to automatically create a reservation for the week starting from the desired start day and ending on the desired end day of the week. The recurring reservation continually generates new reservations. The **week depth** is used to decide when a reservation is created. For example, if a reservation starts 4 weeks from now and the **week depth** is set to 2 weeks the reservation will not be created for 2 more weeks.

Field	Description
<b>Start Day</b>	This field allows the user to select the day and time in the week the reservation begins.
<b>End Day</b>	This field allows the user to select the day and time in the week the reservation ends.
<b>Week Depth</b>	This field allows the user to specify how many weeks in advance Workload Manager should create recurring reservations.

**Infinite Reservation**

A reservation will be created that will continue indefinitely. No start or end time is required.

## Advanced Options

### [Event Triggers](#)

#### Create Trigger 1-6

These fields allow the user the ability to attach triggers to a reservation.

#### Grid Sandboxing

Field	Description
<b>Allow grid sandboxing</b>	By default, this allows only the resources in the recurring reservation to be visible to grid peers.
<b>Cluster List</b>	List of clusters that have access to the grid sandbox.

#### Misc. Options

Field	Description
<b>Partition</b>	Clusters can be divided into different sections. These sections are called partitions. Users can only request one specific partition for each reservation. Consult with your system administrator to learn which partition is best suited for your reservation.
<b>Node Features</b>	Some jobs require a specific feature on a node. A node feature is a custom tag attached to a specific list of nodes. Consult your system administrator for specific information regarding each tag.

## Modify Reservation

### Summary

A reservation is a time frame on the cluster reserved for a particular need. Reservations usually reserve resources such as nodes or processors on the cluster. The Modify Reservation window allows you to view and modify existing reservations.

### [Basic Information](#)

#### Name

This field allows a user to create a name for the reservation. Workload Manager appends a numerical value to the end of the reservation name allowing users the ability to enter duplicate reservation names without affecting any other reservation.

### **Owner**

An owner is a user, group, account, class, or quality of service. A reservation can reserve only the resources that the owner has access to. This field displays the owner of the reservation. If the reservation is a job reservation, this field will be blank.

### **Type**

This field displays whether a reservation is a user or job reservation.

### **Sub Type**

This is the type of reservation. Some examples of the available types are grid, standing reservation, user, maintenance, etc.

### **ACL**

This field displays who is allowed to access resources on the reservation.

### **Statistics**

This field displays the percentage of processors seconds reserved by the reservation that were used by a job or multiple jobs.

## Accounting Credentials

### **User**

This field displays the user ID used by the reservation. If this field is empty no user can directly access this reservation.

### **Group**

This field displays the group ID used by the reservation. If this field is empty no group can directly access this reservation.

### **Account**

This field displays the account ID used by the reservation. If this field is empty no account can directly access this reservation.

### **Class**

This field displays the class ID used by the reservation. If this field is empty no class can directly access this reservation.

### **Quality of Service (QoS)**

This field displays the quality of service (QoS) ID used by the reservation. If this field is empty no quality of service can directly access this reservation.

## Cluster Information

### **Flags**

Cluster Manager schedules reservations differently according to their flags. This field displays the reservation flags.

**Node Expression**

This field displays a list of nodes required by the job to execute. The list of nodes is a regular expression. A node is a computer consisting of 1 or more processors.

**Partition**

This field displays the partition on which the reservation will be configured.

**Required Node Count**

This field displays the number of nodes required by the reservation. A node is a computer consisting of 1 or more processors.

**Required Node List**

This field displays a list of nodes required by the reservation. A node is a computer consisting of 1 or more processors.

**Required Task Count**

This field displays the number of processors required by the reservation.

**Resources**

This field displays what type of resource is reserved by the reservation.

Time Frame**Start Time**

Some jobs require a specific amount of time before they are allowed to start. This field allows the user to define the earliest time the job can start. By default, a job may start as soon as resources become available.

**End Time**

This field displays the time at which the reservation will finish.

**Duration**

The duration is the job's estimated execution time. If a job requires more time than the specified duration, duration violation policies come into effect. Consult your system administrator for more information regarding these policies. If no duration is specified, a default walltime is applied. Consult your system administrator for more information regarding your cluster's default walltime.

Node List**Nodes**

Each button displayed represents a node that the reservation has reserved. When the button is selected, the Modify Node window will appear containing information about the node.

## List Reservations

### Summary

A reservation is a time frame on the cluster reserved for a particular needed. Reservations usually reserve resources such as nodes or processors on the cluster. Reservations are created either by a user or by a job. A reservation created by a user is called a user reservation while a reservation created by a job is called a job reservation. All executing jobs have reservations.

### List Reservation Fields

Field	Description
<b>ACL</b>	This field displays who is allowed to access resources on the reservation.
<b>Account</b>	This field displays which accounts can access this reservation. If this field is blank, no accounts have been given access to the reservation.
<b>Accounts in ACL</b>	This field displays which accounts can access resources on the reservation.
<b>Allocated Nodes</b>	A node is a computer consisting of 1 or more processors. The allocated node list is a list of the nodes that the reservation is using.
<b>Architecture</b>	This field displays the hardware architecture of the reservation. The exact hardware information displayed will depend upon the information the resource manager supplies to Workload Manager.
<b>Class</b>	This field displays which classes can access this reservation. If this field is blank, no classes have been given access to the reservation.
<b>Classes in ACL</b>	This field displays which classes are allowed to access resources on the reservation.
<b>Duration</b>	The duration is the job's estimated execution time. The format used is <code>days:hours:minutes:seconds</code> .
<b>End Time</b>	If the reservation will end in less than 12 hours, the value is displayed in the format of <code>hours:minutes:seconds</code> , where a negative value indicates that the reservation ended that many <code>hours:minutes:seconds</code> ago. A positive value indicates that the reservation will end in that many <code>hours:minutes:seconds</code> . Resting the mouse over the value displays the exact date that the reservation ended or will end. An end time that is years in the future often indicates that the reservation was created without any end time specified and Workload Manager inserted a default end time.

Field	Description
<b>Flags</b>	Cluster Manager schedules reservations differently according to their flags. This field displays the reservation flags.
<b>Global ID</b>	This field only displays information when multiple resource managers are present.
<b>Group</b>	This field displays which groups can access the reservation. If this field is blank, no groups have been given access to the reservation.
<b>Groups in ACL</b>	This field displays which groups can access resources on the reservation.
<b>Max Tasks</b>	This field displays the maximum number of processors a reservation can use.
<b>Memory</b>	This field displays how much memory the reservation requires per task.
<b>Messages</b>	This field allows users the option of adding a message or comment to a reservation.
<b>Name</b>	This field allows users to create a name for the reservation. Workload Manager appends a numerical value to the end of the Reservation ID allowing users the ability to enter duplicate Reservation IDs without affecting any other reservation.
<b>Node Expression</b>	A node is a computer consisting of 1 or more processors. This field displays a list of nodes required by the job to execute. The list of nodes is a regular expression.
<b>Node Set Policy</b>	This field displays the policy that the reservation will use to select the nodes.
<b>OS</b>	This field displays the operating systems the job requires.
<b>Owner</b>	This field displays the owner of the reservation. A reservation can reserve only the resources that the owner has access to. An <i>owner</i> is a user, group, account, class, or quality of service.
<b>Partition</b>	Clusters can be divided into different sections. These sections are called partitions. Users can only request one specific partition for each reservation. Consult your system administrator to learn which partition is the best suited for your reservation.
<b>QoS</b>	This field displays which quality of service (QoS) will be able to access this reservation. If this field is blank, no qualities of service (QoS) have been given access to the reservation.
<b>QoS in ACL</b>	This field displays which qualities of service can access resources on the reservation.

Field	Description
<b>Req. Feature List</b>	A feature is a custom attribute attached to a node. This field displays the node features required for a node to be used for the reservation.
<b>Req. Feature Policy</b>	This field displays the policy that the reservation will use to select the features.
<b>Req. Node Count</b>	A node is a computer consisting of 1 or more processors. This field displays the number of nodes required by the reservation.
<b>Req. Node List</b>	A node is a computer consisting of 1 or more processors. This field displays a list of nodes required by the reservation.
<b>Req. Task Count</b>	This field displays the number of processors required by the reservation.
<b>Resources</b>	This field displays what type of resource is reserved by the reservation.
<b>Spec Name</b>	This field displays information for multiple resource managers.
<b>Start Time</b>	If the reservation will start in less than 12 hours, the value is displayed in the format of hours:minutes:seconds, where a negative value indicates that the reservation will start in that many hours:minutes:seconds. A positive value indicates that the reservation started that many hours:minutes:seconds ago. Resting the mouse over the value will display the exact date that the reservation started or will start. The colored bar shows the percentage of the reservation that has completed. The white space indicates the remaining reservation time.
<b>Statistics</b>	This field displays statistical information relating to the reservation.
<b>Sub Type</b>	This displays the type of reservation. Some examples of the available types are grid, standing reservation, user, maintenance, etc.
<b>Task Count</b>	A task is a group of resources that must all be on the same node. This field displays how many groups of resources will be required to create this reservation.
<b>Timeline</b>	This displays the reservation time lines. The green bar indicates the used amount of the reservation while the blue bar indicates the remaining amount of the reservation. The display options on the left side allow a user change how much of the time line is displayed.
<b>Trigger</b>	This field displays information about any trigger that is attached to the reservation.
<b>Type</b>	This field displays whether a reservation is a user or job reservation.

Field	Description
<b>User</b>	This field displays which users can access this reservation. If this field is blank, no users have been given access to the reservation.
<b>Users in ACL</b>	This field displays which users can access resources on the reservation.

*List Jobs/Job Templates*

You can change the set of columns MCM displays in the Reservations table by clicking **View > Change Column Group** in the main menu bar and selecting one of the following options.

Column Group	Categories Displayed
<b>Summary</b>	Name, Type, ACL, Start Time, End Time, Duration
<b>Credentials</b>	Name, Users in ACL, Groups in ACL, Accounts in ACL, Classes in ACL, QoS in ACL
<b>Time</b>	Name, Type, Start Time, End Time, Duration
<b>Resources</b>	Name, Type, Partition, Resources
<b>Required Resources</b>	Name, Type, Req. Feature List, Req. Feature Policy, Req. Node Count, Req. Node List, Req. Task Count
<b>Flags</b>	Name, Type, Flags
<b>Nodes</b>	Name, Type, Allocated Nodes, Node Expression, Node Set Policy
<b>Statistics</b>	Name, Type, Statistics
<b>Comments</b>	Name, Type, Messages
<b>Tasks</b>	Name, Type, Max Tasks, Req. Task Count, Task Count
<b>Identification</b>	Name, Type, Global ID, Owner
<b>Trigger</b>	Name, Type, Trigger

## List Recurring Reservations

### Summary

A recurring reservation, also referred to as a standing reservation or a reservation generator, creates reservations according to user-defined settings. To choose which fields you would like to view in the chart, click **View > Visible Columns** in the main menu bar. To add fields, create a check mark by clicking in the box to the left of the field you would like to view. To remove fields, click on the checked box.

### List Recurring Reservations

Field	Description
<b>Account</b>	This field displays which accounts can access the created reservation.
<b>Class</b>	This field displays which classes can access the created reservation.
<b>Days</b>	This field displays which days of the week the reservations will start. This field will only display information if the period is set to daily.
<b>Disabled</b>	This field displays whether a particular reservation generator setting has been disabled.
<b>Group</b>	This field displays which groups can access the created reservation.
<b>Host List</b>	The host list is a list of the nodes that the reservation is using. A node is a computer consisting of 1 or more processors.
<b>Name</b>	This field displays the reservation's name.
<b>Owner</b>	A reservation generator can reserve only the resources that the owner can access. This field displays the owner of the reservation generator. An <i>owner</i> is a user, group, account, class, or quality of service.
<b>Period</b>	This field displays the interval in which the reservations will be created. The display options are daily, weekly or infinitely.
<b>QoS</b>	This field displays which quality of service can access the created reservation.
<b>User</b>	This field displays which users can access the created reservation.

## Reservation Timeline

### Summary

The Reservation Timeline window displays each reservation. On the left side of the Now line is the amount of the reservation that has been used. The right side is the remaining amount of the reservation.

## Reservation Calendar

### Summary

The reservation calendar displays reservations color coded by Account. If a reservation does not have an Account, it is shown in gray. The height of a reservation indicates the number of processors it needs. Thus, tall reservations require more processors. Detailed information about the reservation can be seen upon mouseover. To modify a reservation, simply click on it and a reservation modification window will appear.

## Triggers

### List Triggers

#### Summary

Workload Manager can launch events or triggers based on certain events. For example, an administrator may want an email sent when the reservation usage falls below a certain percentage, or a user may want to launch an evaluation script 5 minutes before his or her job is scheduled for completion.

#### List Triggers

You can change the set of columns MCM displays in the Triggers table by clicking **View > Change Column Group** and selecting one of the following options.

Column Group	Columns Displayed
<b>Summary</b>	Trigger ID, Trigger State, Resource ID, Resource Type, Resource Event, Trigger Action, Offset, Threshold
<b>Executable</b>	Script, Flags
<b>Diagnostic</b>	Start Time, Messages, Output File, Error File

When you click **View > Visible Columns...**, you can manually add or remove individual columns in your **Triggers** table. The following table describes each column:

Field	Description
<b>Description</b>	This field displays the description of the trigger if one was written.
<b>Error File</b>	This field displays the location of the file containing all the trigger error messages.
<b>Est. Start Time</b>	This field displays the date and time the trigger will execute if it is possible to calculate it. For example, reservation end times and job completion times can be calculated. Node or scheduler failures cannot be calculated ahead of time.
<b>Flags</b>	This field displays which flags have been set for this trigger.
<b>Messages</b>	This field displays the status information indicating possible failures or unexpected conditions.
<b>Offset</b>	This field displays the number of seconds after a resource event occurs that the trigger will execute. If this value is negative, the trigger will execute that many seconds before the resource event occurs.
<b>Output File</b>	This field displays the location of the file containing all the trigger output messages.
<b>Resource Event</b>	This field displays the event that must occur for the trigger to execute. The possible events are when the resource is created, when the resource starts, when the resource ends, or when a failure occurs in the resource.
<b>Resource ID</b>	This field displays the ID of the job, reservation, or node to which the trigger is attached.
<b>Resource Type</b>	This field displays whether the trigger is attached to a job, reservation, node, or the scheduler.
<b>Script</b>	This field displays the script, application, program, or executable that will be executed when the trigger is starts.
<b>Start Time</b>	This field displays the date and time the trigger started. This field is only populated after the trigger has been executed.
<b>Threshold</b>	This field displays the reservation usage threshold for this trigger. If the reservation falls below the displayed usage, the trigger will execute.
<b>Trigger Action</b>	This field displays the type of trigger action that will occur when the trigger is executed. The possible trigger types are cancel the resource the trigger is attached to; email the administrator; or execute a script, application, program, or executable.

Field	Description
<b>Trigger ID</b>	This field displays the unique ID assigned to the trigger by Workload Manager.
<b>Trigger State</b>	This field displays the execution status of the trigger. If the state is Idle, the trigger is waiting to execute. If the state is Active, the trigger is executing. Once the trigger has executed, the state displayed will be Successful or Failure, depending on the outcome of the trigger action.

## Resources

### Resources Overview

The resources category gives administrators the ability to view, modify, and set policies and attributes while effectively diagnosing various system resources.

## Visual cluster

### Summary

The visual cluster is an easy and concise way of viewing your entire cluster and the status of each node. The table and explanation below explain how to interpret the visual cluster:

Table 4-2: Visual Cluster Example

	Slot 1	Slot 2	Slot 3
<b>Rack 1</b>	Node A		Node D
<b>Rack 2</b>		Node B	
<b>Rack 3</b>			Node C

A rack is a physical frame that holds a node. The slot is the location of the node inside the rack. The racks make up the first column of the table. The slot locations increase from left to right. For example, Node A is located on Rack 1 in Slot 1. Node D is also located on Rack 1 but instead of Slot 1 it's located in Slot 3. In the visual cluster Node A through Node D are displayed as icons. The different icons can represent node state, node attributes, reservations, jobs, and/or nodes. The subpanel sections below describe these states in more detail. Further information can be gathered about nodes by hovering the mouse over any nodes.

It should be noted that the visual cluster is for display purposes only and the location of the node does not play any part in how Workload Manager schedules.

## Node Attribute Selector

The node attribute selector gives the user the power to see various attributes of the nodes displayed in the Visual Cluster. This allows the user to compare and contrast attributes of interest. Node attributes include standard categories such as architecture, OS, hardware metrics (memory, disk, swap, etc.), as well as any metric read in through Moab as a generic metric (a node's GMETRIC). The default display for node attributes is the corresponding color of the outer rim of each node cell. This can be changed in the section titled "Node Display Preferences".

The Clear Attribute button clears any current selection and node attribute.

Once a node attribute is selected, Moab Cluster Manager determines the number of nodes and processors that describe each attribute and displays a corresponding key value that matches the Visual Cluster by color. Each attribute's display can be individually controlled via the check box next to each node attribute name and color.

If the node attribute is a numerical value - for example, a generic metric - then Moab Cluster Manager will attempt to place the values into a reasonable range as to effectively categorize the values.

## Nodes

### Modify Nodes

#### Node Information

##### **Node ID**

This field assigns the node profile to a desired node.

##### **Node State**

This field displays the operating status of the node. For example, unknown, draining, busy, running, down, idle, etc.

#### Node Usage Limits

##### **Max Jobs**

This field allows the user to specify the maximum number of simultaneous jobs allowed to run on this node.

##### **Jobs per User**

This field allows the user to specify the maximum number of simultaneous jobs per end user allowed to run on the node.

### **Max Load**

This field allows the user to specify the maximum percentage of load allowed to run on this node. Load is the number of jobs divided by the number of processors.

### General Attributes

#### **Priority**

This field allows a user to specify the fixed node priority relative to other nodes. The default node priority is 0.

#### **Speed**

This field allows a user to specify the speed as a relative value. A base node generally has a speed of 1.0, but may be changed based on memory, networking interface, etc. It does not have to be proportional to processor speeds.

#### **Processor Speed**

This field allows a user to specify the processor(s) speed on this node. This provides Workload Manager the information needed to schedule nodes with similar processor speeds.

#### **Load**

This field allows the user to specify the maximum percentage of load allowed to run on this node. Load is the number of jobs divided by the number of processors.

#### **Partition**

Clusters can be divided into different sections. These sections are called partitions. This field displays the partition to which the node is assigned.

#### **Rack**

This field allows a user to specify the rack number where the node is located.

#### **Slot**

This field allows a user to specify the slot number where the node is located.

#### **Features**

This field allows a user to specify features assigned to this node.

### Statistics

#### **Active Time**

This field displays the total time the node has actively been executing jobs.

#### **Up Time**

This field displays the total time the node has been available to execute jobs.

#### **Total Time**

This field displays the total time the node has been on the cluster.

## Node Software

### **OS**

The operating system the node uses.

### **OS List**

The list of possible operating systems the node may use.

## Node Attributes

### **Architecture**

This field displays the hardware architecture of the node. The exact hardware information displayed depends on the information the resource manager supplies to Workload Manager.

### **Available Classes**

This field displays classes that can use the node.

### **Node Priority Function**

If Moab is set up to use the **NODEALLOCATIONPOLICY** of *PRIORITY*, you will also see the Node Priority Function Editor panel. This editor allows you to modify the function that is used to assign the node its priority. This function corresponds to the **PRIORITYF** attribute in Moab's configuration.

A valid priority function consists of any number of function expressions. A function expression is a coefficient multiplied by an attribute. For example, one function expression could be 5 \* Available Memory. All the function expressions are summed up to get an effective priority value. This summation is then used to calculate the node's effective priority.

For more information about node allocation policies or the *PRIORITY* policy, see the [Moab documentation](#).

## Over Commit

### **Disk**

Allows overcommitting of the disk space of the node.

### **Memory**

Allows overcommitting of the memory of the node.

### **Processors**

Allows overcommitting of processors of the node.

### **Swap**

Allows overcommitting of the swap space of the node.

## Node Hardware

### **Free Disk Space**

This field displays the available disk space, measured in megabytes (MB), on the node.

**Free Memory**

This field displays the available memory, measured in megabytes (MB), on the node.

**Free Processors**

This field displays the number of available processors on the node.

[Job List](#)

Job List displays all the jobs scheduled on the node.

[Reservation List](#)

Reservation List displays all the reservations on the node.

**List Nodes**

You can change the set of columns MCM displays in the Nodes table by clicking **View > Change Column Group** in the main menu bar and selecting one of the following options.

Default Categories	Columns Displayed
<b>Available Resources</b>	Node ID, State, Available Disk, Available Memory, Available Proc, Available Swap
<b>Comments</b>	Node ID, State, Comments
<b>Configured Resources</b>	Node ID, State, Total Disk, Total Memory, Total Procs, Total Swap
<b>Description</b>	Node ID, State, Features, Rack, Slot, Network, Node Type, O.S., Proc Speed, Size, Speed
<b>Diagnostics</b>	Node ID, State, Load, Messages, Reservation Count, Block Reason, Power
<b>Generic Metrics</b>	Node ID, all configured GMetrics
<b>Maximum Jobs &amp; Procs</b>	Node ID, State, Max. Job, Max. Job Per User, Max. Proc, Max. Proc Per Class
<b>Summary</b>	Node ID, State, Class, Features, Job List, Messages, O.S. List, Total Procs
<b>Usage Limits</b>	Node ID, State, Max I/O In, Max I/O Out, Max. Job, Max. Job Per User, Max. Load, Max. PE Per Job, Max. Proc, Max. Proc Per Class

## List Node Fields

Field	Description
<b>Active Time</b>	This field displays the total time the node has spent doing work.
<b>Architecture</b>	This field displays the hardware architecture of the node. The exact hardware information displayed will depend upon the information the resource manager supplies to Workload Manager.
<b>Available Class</b>	This field displays the classes that can access the node.
<b>Available Consumable Resources</b>	This field displays the restricted resources and the current number available for the node. Some clusters have restrictions placed upon certain nodes. Usually these restrictions are in the form of software licenses. Sometimes a software license can restrict the number of jobs that can simultaneously be using the software on a node. Consumable resources allow a system administrator to define the number of licenses or other restricted resources available on a particular node.
<b>Available Disk</b>	This field displays the available disk space, measured in megabytes (MB), on the node.
<b>Available Memory</b>	This field displays the available memory, measured in megabytes (MB), on the node.
<b>Available Proc</b>	This field displays the number of available processors on the node.
<b>Available Swap</b>	This field displays the available swap, measured in megabytes (MB), on the node.
<b>Block Reason</b>	This field displays any error messages related to the node.
<b>Charge Rate</b>	This field displays the charging rate to the usage of particular resources.
<b>Class</b>	This field displays the classes that can access the node.
<b>Comments</b>	This field allows a user to attach a comment to the node.

Field	Description
<b>Consumable Resources</b>	This field displays the restricted resources and the current number configured for the node. Some clusters have restrictions placed upon certain nodes. Usually these restrictions are in the form of software licenses. Sometimes a software license can restrict the number of jobs that can simultaneously be using the software on a node. Consumable resources allow a system administrator to define the number of licenses or other restricted resources available on a particular node.
<b>Features</b>	A feature is a custom attribute often describing a unique hardware or software configuration associated with the node. This field displays the features associated with the node.
<b>Job List</b>	A node can execute one or more jobs simultaneously. This field displays a list of jobs currently executing on the node.
<b>Load</b>	The load is the number of processors on the node divided by the number of jobs on the node. This field displays what the current node load is.
<b>Max I.O. In</b>	This field displays the maximum disk input in bytes that can occur before the node state is changed to busy.
<b>Max I.O. Load</b>	This field displays the maximum disk input and output in bytes that can occur before the node state is changed to busy.
<b>Max I.O. Out</b>	This field displays the maximum disk output in bytes that can occur before the node state is changed to busy.
<b>Max. Job</b>	This field displays the maximum number of jobs allowed on the node at one time.
<b>Max. Job Per User</b>	This field displays the maximum number of jobs for a single user allowed on the node at one time.
<b>Max. Load</b>	The <i>load</i> is defined as the number of processors on the node divided by the number of jobs on the node. This field displays the maximum load for the node.
<b>Max. PE (Processor Equivalent) Per Job</b>	This field displays the maximum number of processor equivalents* per job allowed on this node at one time.
<b>Max. Proc</b>	This field displays the maximum number of utilized processors allowed on this node at one time.

Field	Description
<b>Max. Proc Per Class</b>	This field displays the maximum number of utilized processors per class allowed on this node at one time.
<b>Messages</b>	This field displays information messages provided by Workload Manager relating to the node.
<b>Network</b>	This field displays the network hardware on the node.
<b>Node ID</b>	All nodes require a unique ID. This field displays that ID.
<b>Node Type</b>	A <i>node type</i> is a custom tag attached to a node. It is usually used in conjunction with an allocation manager such as QBank to assign different charge rates according to the specific node type. This field displays the node type attached to the node.
<b>O.S.</b>	A node is configured with a specific operating system. This field displays the node's configured operating system.
<b>O.S. List</b>	A node is configured with a specific operating system. This field displays the node's configured operating system, as well as other operating systems that are compatible with the configured operating system.
<b>Partition</b>	Clusters can be divided into different sections. These sections are called partitions. This field displays the partition to which the node is assigned.
<b>Power</b>	This field displays whether the node is powered on or off.
<b>Priority</b>	This field displays the priority of the node. The default priority is 0.
<b>Priority Function</b>	This field displays which priority function will be used to calculate a node's priority.
<b>Proc Speed</b>	This field displays the processor speed as gathered from the resource manager.
<b>Rack</b>	This field displays the rack number where the node is logically located.
<b>Reservation Count</b>	This field displays the number of reservations on the node.
<b>Reservations</b>	This field displays reservations currently set on it.
<b>Size</b>	This field displays a description of the size of the node such as 1u or 2u.

Field	Description
<b>Slot</b>	This field displays the slot number where the node is logically located.
<b>Speed</b>	This field displays the speed of the node. A base node has a typical speed of 1.0. A node that is 50% faster has a speed of 1.5.
<b>State</b>	This field displays the operating status of the node. For example, unknown, draining, busy, running, down, idle, etc.
<b>Substate</b>	This field displays a description of the node's state generated by the resource manager.
<b>Total Disk</b>	This field displays the total disk space, measured in megabytes (MB), on the node.
<b>Total Memory</b>	This field displays the total memory, measured in megabytes (MB), on the node.
<b>Total Procs</b>	This field displays the total number of processors on the node.
<b>Total Swap</b>	This field displays the total swap space, measured in megabytes (MB), on the node.
<b>Total Time</b>	This field displays the total time the node has been on the cluster.
<b>Up Time</b>	This field displays the total time the node has been available to execute jobs

## Node calendar

### Summary

The node calendar displays the jobs and reservations on a calendar. Current and future reservations and jobs are shown as well as historical jobs and reservation. The top bar or x-axis is the displayed time frame. The left bar or y-axis is the nodes on the cluster. The colored boxes/cells in the table are identified in the display key.

The node calendar supports 4 time frames. Days in Month, Days in Week, Hours in Day, Minutes In Hour. The top left tabs allow the user to choose the desired time frame. When the **Display Selected Time Frame** button is selected, the currently selected time frame will be displayed.

### Display key

The Display Key panel allows the user to show or hide resources on the node calendar.

The Current Time displayed in green allows a user to see what the current time on the cluster is according to Moab's time reporting.

Jobs displayed in light blue shows all jobs according to the job reservations created for that job.

Reservations displayed in dark blue shows all reservations that fit within the specified time frame.

## Resize calendar

The resize calendar panel, when enabled, reduces the size of each box/cell in the table to allow the user to see more nodes at once. The node names are not visible when the table is compacted.

# Partitions

## Modify a Partition Profile

Clusters can be divided into different sections. These sections are commonly called partitions. A partition is a semi-permanent division of the cluster and is most often used when certain nodes contain unique hardware. It should be recognized that jobs are not allowed to run in more than one partition. If jobs need the ability to span multiple partitions an infinite reservation should be used instead of a partition.

The ability to modify partitions is not currently available in Cluster Manager.

## List Partitions

You can change the set of columns MCM displays in the Partitions table by clicking **View > Change Column Group** in the main menu bar and selecting one of the following options.

Default Categories	Columns Displayed
<b>Disk</b>	ID, Total Disk, Reserved Disk, Utilized Disk
<b>Memory</b>	ID, Total Memory, Reserved Memory, Utilized Memory
<b>Nodes</b>	ID, Total Nodes, Reserved Nodes, Utilized Nodes
<b>Processors</b>	ID, Total Processors, Reserved Processors, Utilized Processors
<b>Summary</b>	ID, Resource Manager
<b>Swap</b>	ID, Total Swap, Reserved Swap, Utilized Swap

## List Partition Fields

When you click **View > Visible Columns...** in the main menu bar, you can manually add or remove individual columns in your **Partitions** table. The following table describes each column:

Field	Description
<b>ID</b>	This field displays the partition ID, or the name of the partition.
<b>Reserved Disk</b>	This field displays the amount of disk space reserved by this partition. The disk space is measured in megabytes (MB).
<b>Reserved Memory</b>	This field displays the amount of memory reserved in this partition. The memory is measured in megabytes (MB).
<b>Reserved Nodes</b>	This field displays the number of nodes reserved in this partition.
<b>Reserved Processors</b>	This field displays the number of procs reserved in this partition.
<b>Reserved Swap</b>	This field displays the amount of swap space reserved by this partition. The swap space is measured in megabytes (MB).
<b>Resource Manager</b>	This field displays the resource manager ID of which this partition is a member.
<b>Total Disk</b>	This field displays the total disk space available in the partition. The disk space is measured in megabytes (MB).
<b>Total Memory</b>	This field displays the total memory available in this partition. The memory is measured in megabytes (MB).
<b>Total Nodes</b>	This field displays the total number of nodes available in this partition.
<b>Total Processors</b>	This field displays the total number of processors available in this partition.
<b>Total Swap</b>	This field displays the total swap space available in the partition. The swap space is measured in megabytes (MB).
<b>Utilized Disk</b>	This field displays the amount of disk space currently being used by this partition. The disk space is measured in megabytes (MB).

Field	Description
<b>Utilized Memory</b>	This field displays the amount of memory currently being used by this partition. The memory is measured in megabytes (MB).
<b>Utilized Nodes</b>	This field displays the number of nodes currently being used in this partition.
<b>Utilized Processor</b>	This field displays the number of processors currently being used in this partition.
<b>Utilized Swap</b>	This field displays the amount of swap space currently being used by this partition. The swap space is measured in megabytes (MB).

## Scheduler settings

This window provides a control center for the basic operations of Workload Manager.

### Control panel

Button	Description
<b>Pause/Resume</b>	The Pause button stops Workload Manager from scheduling any new jobs but will not turn Workload Manager off. The resume button, which replaces the pause button when Workload Manager is paused, will allow Workload Manager to begin scheduling jobs again.
<b>Restart</b>	The Restart button shuts down Moab and restarts it using the original execution environment and command line arguments.
<b>Shut Down</b>	The Shut Down button turns Workload Manager off. Please note that Workload Manager cannot be restarted from Cluster Manager.
<b>Stop Iterations</b>	The Stop Iterations button causes Workload Manager to stop scheduling once it reaches the iteration defined in the field. The iteration is defined as the cycle that Workload Manager is currently on. When Workload Manager starts its cycle is 0. Workload Manager increases the cycle by one approximately every 30 seconds.

### [System settings](#)

This window provides an administrator the option of changing numerous Workload Manager settings.

**Name**

This field allows an administrator to name the cluster. The name is only available for administrator convenience and is not used by Workload Manager.

**Host**

This field allows an administrator to define the host name that the Workload Manager subcomponents or clients use to connect to Workload Manager.

**Port**

This field allows an administrator to define the port that the Workload Manager subcomponents or clients use to connect to Workload Manager.

**Home Directory**

This field allows an administrator to define the directory where Workload Manager's configuration, statistics, and log files are located.

**Feedback Program**

This field allows an administrator to define a program that will be run at the completion of each job. Usually the program is used to contact the user through email informing him/her that the job completed execution.

**Notify Program**

This field allows an administrator to define a program that will be run when messages or alerts occur in Workload Manager.

**RM Poll Interval**

This field is the time in between which Workload Manager will communicate with the resource manager.

**Node Purge Time**

This field allows an administrator to define the amount of time Workload Manager will keep track of a node which is no longer reported by the resource manager. This value should be increased when using a resource manager that often loses information about a node due to internal failures.

**Job Purge Time**

This field allows an administrator to define the amount of time Workload Manager will keep track of a job which is no longer reported by the resource manager. This value should be increased when using a resource manager that often loses information about a job due to internal failures.

**[Log Settings](#)**

Logging is defined as recording error, diagnostic, and informational messages to a file. This window allows an administrator to configure the logging that occurs in Workload Manager.

### **Log Level**

This field allows an administrator the option of specifying the amount of data recorded in the log files. A value of 1 means almost no data is recorded while a value of 9 means all the data is recorded. Each value increment means that approximately double the amount of data is logged to the log files. The default log level is 3.

**Log Facilities**

This field determines what is recorded in the log file.

Setting	Description
<b>fALL</b>	This option records all the events that occur.
<b>fBANK</b>	This option records messages that involve QBank.
<b>fCKPT</b>	This option records messages that involve the checkpoint file.
<b>fCONFIG</b>	This option records messages that involve the configuration file.
<b>fCore</b>	This option records Workload Manager core messages.
<b>fFS</b>	This option records messages that involve fairshare.
<b>fLL</b>	This option records messages that come from LoadLeveler Resource Manager.
<b>fPBS</b>	This option records messages that come from the Torque/OpenPBS Resource Manager.
<b>fRM</b>	This option records resource manager messages.
<b>fSched</b>	This option records messages that involve the scheduler.
<b>fSIM</b>	This option records messages that occur during the simulation operating mode.
<b>fSock</b>	This option records messages that involve the socket communication.
<b>fSTAT</b>	This option records messages that involve statistics.
<b>fSTRUCT</b>	This option records messages that involve Workload Manager's structure.
<b>fUI</b>	This option records messages that involve the user interface.
<b>fWIKI</b>	This option records messages that involve WIKI.
<b>fSDR</b>	This option records messages that involve system data repository.

**Log Directory**

This field allows an administrator to specify the directory in which log files will be maintained.

**Log File**

This field allows an administrator to specify the name of the Workload Manager log file.

**Log File Max Size**

This field allows an administrator the option of specifying the maximum allowed size (in bytes) of the log file before it will be rolled.

**Log File Roll Depth**

When a log file reaches its maximum size, it is *rolled* or renamed to another filename and a new log file is created using the original file name. This field allows an administrator the option of defining the number of renamed files Workload Manager should maintain.

Statistics

Credential statistics are disabled by default although cluster-wide statistics are always enabled.

**Enable Credentials Statistics**

This section allows an administrator to enable or disable user, group, account, class, or quality of service statistics. Because statistics increase Workload Manager's memory usage, an administrator can decrease the memory footprint of Workload Manager by disabling credential statistics.

**Enable Node Statistics**

This section gives you the option of enabling node statistics for all nodes.

**Number of intervals in each day**

Workload Manager combines statistics into intervals. The Daily Statistical Count allows an administrator the option of increasing or decreasing the number of intervals in each day. A higher amount of intervals creates more precise statistics, but Workload Manager uses more memory when intervals are higher.

## Credentials

### Credentials Overview

The credentials section allows an administrator to view all credentials in the system and their various roles. New credential profiles can be added as well as modified.

### All credentials

This window displays all of the information regarding users, groups, accounts, classes, and qualities of service (QoS) commonly called credentials.

You can change the set of columns MCM displays in the **Credentials** table by clicking **View > Change Column Group** in the main menu bar and selecting one of the following options.

Default Categories	Columns Displayed
<b>Comments &amp; Email</b>	Credential, ID, Comments, E-mail Address
<b>Credits</b>	Credential, ID, Credits, Used Credits
<b>Default Credentials</b>	Credential, ID, User Default, Group Default, Class Default, Account Default, Default QoS
<b>Default Resources</b>	Credential, ID, Partition Default
<b>Fairshare</b>	Credential, ID, Fairshare Type, Fairshare Target
<b>Hard Max Limits</b>	Credential, ID, Max Job, Max Proc, Max Nodes, Max Proc Sec
<b>Membership</b>	Credential, ID, User, Group, Class, Account, QoS
<b>Partition &amp; Reservation</b>	Credential, ID, Partition, Reservation
<b>Priority</b>	Credential, ID, Priority
<b>Utilized Resources</b>	Credential, ID, Utilized Job, Utilized Proc, Utilized Nodes, Utilized Proc Sec
<b>Soft Max Limits</b>	Credential, ID, Soft Max Job, Soft Max Proc, Soft Max Nodes, Soft Max Proc Sec
<b>Statistics</b>	Credential, ID, Enable Statistics

## List Credential Fields

When you click **View > Visible Columns...** in the main menu bar, you can manually add or remove individual columns in your **Credentials** table. The following table describes each column:

Field	Description
<b>Access Resources QT Threshold</b>	This field displays the number of minutes that an idle job must wait before it can access any reservations owned by the Quality of Service.
<b>Access Resources XF Threshold</b>	This field displays the expansion factor value that an idle job must be greater than or equal to in order to access any reservations owned by the Quality of Service.

Field	Description
<b>Account</b>	This field displays the accounts that this particular credential ID can access.
<b>Account Default</b>	This field displays the account that will be used by this credential ID's job if no account is specified.
<b>Class</b>	This field displays the classes that this particular credential ID can access.
<b>Class Default</b>	This field displays the class that will be used by this credential ID's job if no class is specified.
<b>Comments</b>	This field allows a user to enter any comments relating to the Credential.
<b>Create Reservation QT Threshold</b>	This field displays the number of minutes that an idle job must wait before a job reservation will be created for it. A job reservation will guarantee it specific resources, as well as a specific start time.
<b>Create Reservation XF Threshold</b>	This field displays the expansion factor value that an idle job must be greater than or equal to before a job reservation will be created for it. A job reservation guarantees it specific resources, as well as a specific start time.
<b>Credential</b>	This field displays whether the credential is a user, group, account, class, or quality of service (QoS).
<b>Credits</b>	This field displays the total credits available to the credential ID.
<b>Dedicated Resource Cost</b>	What Workload Manager charges for each resource unit dedicated - whether used or not - to a job.
<b>Default QoS</b>	The default quality of service (QoS) is the quality of service (QoS) that will be used by this credential ID's job if no quality of service (QoS) is specified
<b>E-Mail Address</b>	This field allows a user to enter the e-mail address for the Credential.
<b>Enable Statistics</b>	Statistics are tracked for each credential ID. This field allows the user the option of enabling/disabling statistics for each credential ID.

Field	Description
<b>Fairshare Target</b>	Refer to the fairshare section for information regarding fairshare target.
<b>Fairshare Type</b>	Refer to the fairshare section for information regarding fairshare type.
<b>Group</b>	The operating system is usually responsible for the creation of groups. This field displays the groups that this particular credential ID can access.
<b>Group Default</b>	This field displays the group that will be used by this credential ID's job if no group is specified.
<b>ID</b>	All credentials must have an identification unique to its credential type. This field displays the credential identification.
<b>Max Job</b>	The default maximum job is the maximum job value that will be used by this credential ID's job if no maximum job is specified.
<b>Max Nodes</b>	This field displays the maximum nodes value that will be used by this credential ID's job if no maximum nodes value is specified.
<b>Max Proc</b>	This field displays the maximum processors value that will be used by this credential ID's job if no maximum processor is specified.
<b>Max Proc Sec</b>	This field displays the maximum processor seconds value that will be used by this credential ID's job if no maximum processor seconds value is specified.
<b>Partition</b>	Clusters can be divided into different sections. These sections are commonly called <i>partitions</i> . This field displays the partitions this credential ID can access.
<b>Partition Default</b>	This field displays the partition that will be used by this credential ID's job if no partition is specified.
<b>Preemption QT Threshold</b>	This field displays the number of minutes that an idle job must wait before it will be given preemptor access.
<b>Preemption XF Threshold</b>	This field displays the expansion factor value that an idle job must be greater than or equal to before it will be given preemptor access.

Field	Description
<b>Priority</b>	This field displays the priority of the credential.
<b>QoS</b>	This field displays the quality of services (QoS) that this particular credential ID can access.
<b>QoS Flags</b>	This field displays the quality of service (QoS) settings for this credential ID.
<b>Queue Time Weight</b>	This field displays the quality of service weight factor. If an idle job is submitted to this quality of service, the number of minutes that it has been in the queue is multiplied by this value. This increases the job's start priority.
<b>Reservation</b>	The required reservation that any job submitted has to use.
<b>Soft Max Job</b>	The soft maximum job limit restricts the number of jobs allowed to execute for this credential ID. If, however, additional resources are available after all the soft maximum job limits are met, then the hard maximum job limits are used.
<b>Soft Max Nodes</b>	The soft maximum node limit restricts the number of nodes used by any job for this credential ID. If, however, additional resources are available after all the soft maximum node limits are met, then the hard maximum node limits are used.
<b>Soft Max Proc</b>	The soft maximum processor limit restricts the number of processors used by any job for this credential ID. If, however, additional resources are available after all the soft maximum processor limits are met, then the hard maximum processor limits are used.
<b>Soft Max Proc Sec</b>	The soft maximum processor seconds limit restricts the number of processor seconds used by any job for this credential ID. If, however, additional resources are available after all the soft maximum processor seconds limits are met, then the hard maximum processor seconds limits are used.
<b>Used Credits</b>	This field displays the credits used by this credential ID.
<b>User</b>	This field displays the users that this particular credential ID can access.
<b>User Default</b>	This field displays the user that will be used by this credential ID's job if no user is specified.
<b>Utilized Job</b>	This field displays the number of jobs currently executing for this credential ID.
<b>Utilized Nodes</b>	This field displays the number of nodes currently being used by this credential ID's jobs.

Field	Description
<b>Utilized Proc</b>	This field displays the number of processors currently being used by this credential ID's jobs.
<b>Utilized Proc Sec</b>	This field displays the total number of processors used by executing jobs for this particular credential ID times the number of seconds each processor has been used.
<b>Utilized Resource Cost</b>	This field displays what Workload Manager charges for each resource unit consumed/utilized by a job.
<b>XFactor Weight</b>	This field displays the quality of service weight factor. If an idle job is submitted to this quality of service, its expansion factor is multiplied by this value. This increases the job's start priority.

## Visual credentials

This window allows a user to view which credentials can access which credentials. The arrows symbolize that the credential can access the other credential. There are three distinct sections displayed in the window. The first section displays all the credentials that can access the second section. The second section contains only the selected credential. The arrows from the second section to the third section show what the selected credential can access. The first and third sections may not be displayed if they do not contain credentials that are accessible to the credential in the second section.

### Display All Credentials

This field allows a user to display all of the credentials of a specific credential type.

### Display Listed Credentials

This field allows a user to display only the listed credentials.

### Display who can access the selected credential

This field allows a user to specify whether the credentials that access the selected credentials should be displayed.

### Display what the selected credential can access

This field allows a user to specify whether the credentials that the selected credential can access should be displayed.

## Create/Modify a user profile

Users are created by the operating system while user profiles are created by Workload Manager. When a user submits a job, that user becomes visible to Workload Manager and at that moment a credential profile is automatically created for the user.

To create a new user, you must click **Actions > Create User...** in the main menu bar. The following tables describe the fields in the pop-up box. To modify a user, you must double-click the user profile in the Users table (located in the Credentials folder of the Navigation pane).

To control user rights, see the [User access on page 2523](#) page.

## Credential Access

### **User Name**

This field allows an administrator to define the name of the user. Usually this is the user's login name.

### **Group Access List**

The group access is defined by the operating system and cannot be defined by Workload Manager.

### **Account Access List**

This field allows an administrator to define which accounts this user can access.

### **Default Account**

This field allows an administrator to define which accounts will be automatically used if the user doesn't specify an account.

### **QoS (Quality of Service) Access List**

This field allows an administrator to define which qualities of service (QoS) this user can access.

### **Default QoS (Quality of Service)**

This field allows an administrator to define which quality of service (QoS) will automatically be used if the user doesn't specify a quality of service (QoS).

## Resource Access

### **Partitions**

This field allows an administrator to define which partitions this user can access.

### **Default Partition**

This field allows an administrator to define which partitions this user can access by default.

### **Reservation**

This field allows an administrator to define which reservation this user can access.

## Job Usage Limits

### **Idle Max Jobs**

This field allows an administrator the option of setting the user's maximum number of simultaneously idle jobs.

**Max Jobs**

This field allows an administrator to set the user's maximum number of simultaneously executing jobs.

**Max Procs**

This field allows an administrator to set the user's maximum number of simultaneously utilized processors.

**Max Proc Seconds**

This field allows an administrator to set the user's maximum number of simultaneously utilized processor seconds. Processor seconds is the number of processors utilized times the number of seconds they are utilized.

**Max Nodes**

This field allows an administrator to set the user's maximum number of simultaneously utilized nodes. A node is a computer consisting of 1 or more processors.

**Max Wallclock**

This field allows an administrator to set the user's maximum wallclock time available when submitting a job.

**Min Wallclock**

This field allows an administrator to set the user's minimum wallclock time when submitting a job.

**Default Wallclock**

This field allows an administrator to set the user's default wallclock time set when submitting a job with no specified wallclock.

Fairshare & Priority

**Fairshare Policies**

Fairshare is a method of enforcing cluster sharing between credentials. A *credential* is a user, group, account, class/queue, or quality of service (QoS). Fairshare tracks each credential's usage for a desired amount of time and decreases a job's start priority if the fairshare policy is violated. By decreasing a job's start priority, a user specifies that the job will wait longer in the queue before it starts allowing other jobs to execute first.

Policy	Description
<b>Fairshare Floor Policy</b>	If the user's cluster usage is below the fairshare target, then the user's start priority for the job will increase. The user's cluster usage is measured as the total percentage amount of the cluster used by the user.
<b>Fairshare Target Policy</b>	If the user's cluster usage is above or below the fairshare target, then the user's start priority for the job will increase or decrease accordingly. The user's cluster usage is measured as the total percentage amount of the cluster used by the user.
<b>Fairshare Ceiling Policy</b>	If the user's cluster usage is above the fairshare target, then the user's start priority for the job decreases. The user's cluster usage is the total percentage of the cluster used by the user.

**Target**

This field allows an administrator to define the fairshare target for this user. Refer to the fairshare policy for an understanding of how fairshare target will be used.

**User Priority**

This field allows an administrator to define a user's job priority. A user's job priority will increase or decrease the start priority of this user's jobs. Workload Manager, with some exceptions, will start the jobs with the highest start priority first.

Comments

**Comments**

This field allows an administrator to enter comments regarding the user.

Email Address

**Email Address**

This field allows an administrator to add a user's email address to Workload Manager. The email address is only for contact information and is not used by Workload Manager or the resource manager.

Usage Statistics

This is only visible if a profile is being modified.

**System Utilization**

Shows what percent of the entire system has recently been dedicated to this credential. Each value in this line graph represents the number of dedicated processor seconds used by this credential during the profiling duration divided by the number of dedicated processor seconds in the entire system for the profiling duration. The number of dedicated processor seconds for the entire system for a profiling duration is equal to the size of the profiling duration in seconds multiplied by the number of processors.

## Create/Modify a group profile

Groups are created by the operating system while group profiles are created by Workload Manager. When a user submits a job, that user's group becomes visible to Workload Manager and at that moment, a credential profile is automatically created for the group.

To create a new group, you must click **Actions > Create Group...** in the main menu bar. The following tables describe the fields in the pop-up box. To modify a group, you must double-click the group profile in the Groups table (located in the Credentials folder of the Navigation pane).

Credential Access**Group Name**

This field allows an administrator to define the identification name of the group. Usually this is the login name for the group.

**User Access List**

This field allows an administrator to define which users can access this group.

**Account Access List**

This field allows an administrator to define which accounts this group can access.

**Default Account**

This field allows an administrator to define which account is the default of this group.

**QoS (Quality of Service) Access List**

This field allows an administrator to define which qualities of service (QoS) this group can access.

**Default QoS (Quality of Service)**

This field allows an administrator to define which quality of service (QoS) will automatically be used if the group doesn't specify a quality of service (QoS).

Resource Access**Partitions**

This field allows an administrator to define which partitions this group can access.

**Default Partition**

This field allows an administrator to define which partitions this group can access by default.

### **Reservation**

This field allows an administrator to define which reservation this group can access.

### Job Usage Limits

#### **Idle Max Jobs**

This field allows an administrator the option of setting the group's maximum number of simultaneously idle jobs.

#### **Max Jobs**

This field allows an administrator to set the group's maximum number of simultaneously executing jobs.

#### **Max Procs**

This field allows an administrator to set the group's maximum number of simultaneously utilized processors.

#### **Max Proc Seconds**

This field allows an administrator to set the group's maximum number of simultaneously utilized processor seconds. Processor seconds is the number of processors utilized times the number of seconds they are utilized.

#### **Max Nodes**

This field allows an administrator to set the group's maximum number of simultaneously utilized nodes. A node is a computer consisting of 1 or more processors.

#### **Max Wallclock**

This field allows an administrator to set the group's maximum wallclock time available when submitting a job.

#### **Min Wallclock**

This field allows an administrator to set the group's minimum wallclock time when submitting a job.

#### **Default Wallclock**

This field allows an administrator to set the group's default wallclock time set when submitting a job with no specified wallclock.

Fairshare & Priority

**Fairshare Policy**

Fairshare is a method of enforcing cluster sharing between credentials. A credential is a user, group, account, class, or quality of service (QoS). Fairshare tracks each credential's usage for a desired amount of time and decreases a job's start priority if the fairshare policy is violated. By decreasing a job's start priority, the job will wait longer in the queue before it starts, allowing other jobs to execute first.

Policy	Description
<b>Fairshare Floor Policy</b>	If the group's cluster usage is below the fairshare target then the group's start priority for the job increases. The group's cluster usage is measured as the total percentage amount of the cluster used by the group.
<b>Fairshare Target Policy</b>	If the group's cluster usage is above or below the fairshare target then the group's start priority for the job increases or decreases accordingly. The group's cluster usage is measured as the total percentage of the cluster used by the group.
<b>Fairshare Ceiling Policy</b>	If the group's cluster usage is above the fairshare target then the group's start priority for the job decreases. The group's cluster usage is the total percentage of the cluster used by the group.

**Target**

This field allows an administrator to define the fairshare target for this group. Refer to the Fairshare Policy for an understanding of how fairshare target will be used.

**Group Priority**

This field allows an administrator to define a group's job priority. A group's job priority will either increase or decrease the start priority of this group's jobs. Workload Manager, with some exceptions, will start the jobs with the highest start priority first.

Comments

**Comments**

This field allows an administrator to enter any comments regarding the group.

Usage Statistics

This is only visible if a profile is being modified.

### System Utilization

Shows what percent of the entire system has recently been dedicated to this credential. Each value in this line graph represents the number of dedicated processor seconds used by this credential during the profiling duration divided by the number of dedicated processor seconds in the entire system for the profiling duration. The number of dedicated processor seconds for the entire system for a profiling duration is equal to the size of the profiling duration in seconds multiplied by the number of processors.

## Create/Modify an account profile

Account creation occurs in Workload Manager.

To create a new account, you must click **Actions > Create Account...** in the main menu bar. The following tables describe the fields in the pop-up box. To modify an account, you must double-click the account profile in the Accounts table (located in the Credentials folder of the Navigation pane).

### Credential Access

#### Account Name

This field allows an administrator to define the name of the account. This is usually the login name for the account.

#### User Access List

This field allows an administrator to define which users can access this account.

#### Group Access List

This field allows an administrator to define which groups can access this account.

#### QoS (Quality of Service) Access List

This field allows an administrator to define which qualities of service (QoS) this account can access.

#### Default QoS (Quality of Service)

This field allows an administrator to define which quality of service (QoS) will automatically be used if the account doesn't specify a quality of service (QoS).

### Resource Access

#### Partition

This field allows an administrator to define which partitions this account can access.

#### Default Partition

This field allows an administrator to define which partitions this account can access by default.

#### Reservation

This field allows an administrator to define which reservation this account can access.

## Job Usage Limits

### **Idle Max Jobs**

This field allows an administrator the option of setting the account's maximum number of simultaneously idle jobs.

### **Max Jobs**

This field allows an administrator to set the account's maximum number of simultaneously executing jobs.

### **Max Procs**

This field allows an administrator to set the account's maximum number of simultaneously utilized processors.

### **Max Proc Seconds**

This field allows an administrator to set the account's maximum number of simultaneously utilized processor seconds. Processor seconds is defined as the number of processors utilized times the number of seconds they are utilized.

### **Max Nodes**

This field allows an administrator to set the account's maximum number of simultaneously utilized nodes. A node is a computer consisting of 1 or more processors.

### **Max Wallclock**

This field allows an administrator the option of setting the account's maximum wallclock time available when submitting a job.

### **Min Wallclock**

This field allows an administrator the option of setting the account's minimum wallclock time when submitting a job.

### **Default Wallclock**

This field allows an administrator the option of setting the account's default wallclock time set when submitting a job with no specified wallclock.

## Fairshare & Priority

### Fairshare Policies

Fairshare is a method of enforcing cluster sharing between credentials. A credential is a user, group, account, class, or quality of service (QoS). Fairshare tracks each credential's usage for a desired amount of time and decreases a job's start priority if the fairshare policy is violated. By decreasing a job's start priority, the job will wait longer in the queue before it starts, allowing other jobs to execute first.

Policy	Description
<b>Fairshare Floor Policy</b>	If the account's cluster usage is below the fairshare target, then the account's start priority for the job will increase. The account's cluster usage is the total percentage of the cluster used by the account.
<b>Fairshare Target Policy</b>	If the account's cluster usage is above or below the fairshare target, then the account's start priority for the job increases or decreases accordingly. The account's cluster usage is measured as the total percentage amount of the cluster used by the account.
<b>Fairshare Ceiling Policy</b>	If the account's cluster usage is above the fairshare target, then the account's start priority for the job increases or decreases. The account's cluster usage is the total percentage of the cluster used by the account.

### Target

This field allows an administrator to define the fairshare target for this account. Refer to the Fairshare Policy for an understanding of how fairshare target will be used.

### Account Priority

This field allows an administrator to define an account's job priority. An account's job priority increases or decreases the start priority of this account's jobs. Workload Manager, with some exceptions, starts the jobs with the highest start priority first.

### Comments

#### Comments

This field allows an administrator to enter comments regarding the account.

### Usage Statistics

This is only visible if a profile is being modified.

### System Utilization

Shows what percent of the entire system has recently been dedicated to this credential. Each value in this line graph represents the number of dedicated processor seconds used by this credential during the profiling duration divided by the number of dedicated processor seconds in the entire system during the profiling duration. The number of dedicated processor seconds for the entire system for a profiling duration is equal to the size of the profiling duration in seconds multiplied by the number of processors.

## Create/Modify a class profile

Classes are created by the resource manager while class profiles are created by Workload Manager.

To create a new class, you must click **Actions > Create Class...** in the main menu bar. The following tables describe the fields in the pop-up box. To modify a class, you must double-click the class profile in the Class table (located in the Credentials folder of the Navigation pane).

### Credential Access

#### **Class Name**

This field allows an administrator to define the name of the class. Usually this is the login name for the class.

#### **User Access List**

This field allows an administrator to define which users can access this class.

#### **Group Access List**

This field allows an administrator to define which groups can access this class.

#### **Account Access List**

This field allows an administrator to define which accounts this class can access.

#### **Default Account**

This field allows an administrator to define which accounts will automatically be used if the class doesn't specify an account.

#### **QoS (Quality of Service) Access List**

This field allows an administrator to define which qualities of service (QoS) this class can access.

#### **Default QoS (Quality of Service)**

This field allows an administrator to define which quality of service (QoS) will automatically be used if the class doesn't specify a quality of service (QoS).

### Resource Access

#### **Partitions**

This field allows an administrator to define which partitions this class can access.

**Default Partition**

This field allows an administrator to define which partitions this class can access by default.

**Reservation**

This field allows an administrator to define which reservation this class can access.

Job Usage Limits**Idle Max Jobs**

This field allows an administrator to set the class's maximum number of simultaneously idle jobs.

**Max Jobs**

This field allows an administrator to set the class's maximum number of simultaneously executing jobs.

**Max Procs**

This field allows an administrator to set the class's maximum number of simultaneously utilized processors.

**Max Proc Seconds**

This field allows an administrator to set the class's maximum number of simultaneously utilized processor seconds. Processor seconds is the number of processors utilized times the number of seconds they are utilized.

**Max Nodes**

This field allows an administrator to set the class's maximum number of simultaneously utilized nodes. A node is a computer consisting of 1 or more processors.

**Max Wallclock**

This field allows an administrator to set the class's maximum wallclock time available when submitting a job.

**Min Wallclock**

This field allows an administrator to set the class's minimum wallclock time when submitting a job.

**Default Wallclock**

This field allows an administrator to set the class's default wallclock time set when submitting a job with no specified wallclock.

## Fairshare & Priority

### **Fairshare Policies**

Fairshare is a method of enforcing cluster sharing between credentials. A *credential* is a user, group, account, class, or quality of service (QoS). Fairshare tracks each credential's usage for a desired amount of time and decreases a job's start priority if the fairshare policy is violated. By decreasing a job's start priority, the job will wait longer in the queue before it starts, allowing other jobs to execute first.

Policy	Description
<b>Fairshare Floor Policy</b>	If the class's cluster usage is below the fairshare target, then the class's start priority for the job increases. The class's cluster usage is the total percentage of the cluster used by the class.
<b>Fairshare Target Policy</b>	If the class's cluster usage is above or below the fairshare target, then the class's start priority for the job increases or decreases accordingly. The class's cluster usage is the total percentage of the cluster used by the class.
<b>Fairshare Ceiling Policy</b>	If the class's cluster usage is above the fairshare target, then the class's start priority for the job decreases. The class's cluster usage is the total percentage of the cluster used by the class.

### **Target**

This field allows an administrator to define the fairshare target for this class. Refer to the Fairshare Policy for an understanding of how fairshare target will be used.

### **Class Priority**

This field allows an administrator to define a class's job priority. A class's job priority will increase or decrease the start priority of this class's jobs. Workload Manager, with some exceptions, will start the jobs with the highest start priority first.

## Comments

### **Comments**

This field allows an administrator to enter comments regarding the class.

## Usage Statistics

This is only visible if a profile is being modified.

**System Utilization**

Shows what percent of the entire system has recently been dedicated to this credential. Each value in this line graph represents the number of dedicated processor seconds used by this credential for the profiling duration divided by the number of dedicated processor seconds in the entire system for the profiling duration. The number of dedicated processor seconds for the entire system for a profiling duration is equal to the size of the profiling duration in seconds multiplied by the number of processors.

## Create/Modify a quality of service profile

Qualities of service are created by the operating system while quality of service profiles are created by Workload Manager. When a quality of service submits a job then that quality of service becomes visible to Workload Manager and at that moment a credential profile is automatically created for the quality of service.

To create a new QoS, you must click **Actions > Create QoS...** in the main menu bar. The following tables describe the fields in the pop-up box. To modify a QoS, you must double-click the QoS profile in the QoS table (located in the Credentials folder of the Navigation pane).

Credential Access**Quality of Service Name**

This field allows an administrator to define the name of the quality of service. Usually this is the login name for the quality of service.

**User Access List**

This field allows an administrator to define which users can access this quality of service.

**Group Access List**

The group access is defined by the operating system and cannot be defined by Workload Manager.

**Account Access List**

This field allows an administrator to define which accounts this quality of service can access.

**Default Account**

This field allows an administrator to define which accounts will automatically be used if the quality of service doesn't specify an account.

Partitions & Reservations**Partitions**

This field allows an administrator to define which partitions this quality of service can access.

**Default Partition**

This field allows an administrator to define which partitions this user can access by default.

**Reservation**

This field allows an administrator to define which reservations jobs that access this quality of service must use.

Fairshare

**Fairshare Policies**

Fairshare is a method of enforcing cluster sharing between credentials. A credential is a user, groups, account, class/queue, or quality of service (QoS). It consists of tracking each credentials usage for a desired amount of time and decreasing a jobs start priority if the fairshare policy is violated. By decreasing a job's start priority the job will wait longer in the queue before it starts allowing other jobs to execute first.

Policy	Description
<b>Fairshare Floor Policy</b>	If the quality of services cluster usage is below the fairshare target the quality of services start priority for the job increases. The quality of services cluster usage is the total percentage of the cluster used by the quality of service.
<b>Fairshare Target Policy</b>	If the quality of services cluster usage is above or below the fairshare target, the quality of services start priority for the job increases or decreases accordingly. The quality of services cluster usage is the total percentage of the cluster used by the quality of service.
<b>Fairshare Ceiling Policy</b>	If the quality of services cluster usage is above the fairshare target, the quality of services start priority for the job decreases. The quality of services cluster usage is the total percentage of the cluster used by the quality of service.

**Target**

This field allows an administrator to define the fairshare target for this quality of service. Refer to the Fairshare Policy for an understanding of how fairshare target will be used.

Comments

**Comments**

This field allows an administrator to enter comments regarding the quality of service.

Job Usage Limits

**Idle Max Jobs**

This field allows an administrator to set the quality of services maximum number of simultaneously idle jobs.

**Max Jobs**

This field allows an administrator to set the quality of services maximum number of simultaneously executing jobs.

**Max Procs**

This field allows an administrator to set the quality of services maximum number of simultaneously utilized processors.

**Max Proc Seconds**

This field allows an administrator to set the quality of services maximum number of simultaneously utilized processor seconds. Processor seconds is the number of processors utilized times the number of seconds they are utilized.

**Max Nodes**

This field allows an administrator to set the quality of services maximum number of simultaneously utilized nodes. A node is a computer consisting of 1 or more processors.

**Max Wallclock**

This field allows an administrator to set the quality of services maximum wallclock time available when submitting a job.

**Min Wallclock**

This field allows an administrator to set the quality of services minimum wallclock time when submitting a job.

**Default Wallclock**

This field allows an administrator to set the quality of services default wallclock time set when submitting a job with no specified wallclock.

**Charging****Dedicated Cost**

The cost to have dedicated access to this resource, regardless of whether it is being utilized.

**Utilized Cost**

The cost to utilize this resource.

**Prioritization****Queue Time Weight**

This field displays the quality of service weight factor. If an idle job is submitted to this quality of service the number of minutes that it has been in the queue will be multiplied by this value. This increases the jobs start priority.

**Expansion Factor Weight**

This field displays the quality of service weight factor. If an idle job is submitted to this quality of service it's expansion factor is multiplied by this value. This increases the job's start priority.

**Quality of Service Priority**

This field affects the priority given to jobs with this quality of service applied.

## Quality of Service Flags

### **Deadline**

This option makes any job with this quality of service with a completion deadline to finish before that deadline.

### **Dedicated**

This option makes any job submitted to this quality of service to require a dedicated node. A dedicated node is a node that is completely reserved for only one job.

### **Enable User Reservation**

This option makes any user that is a member of this quality of service able to create user/personal reservations.

### **Ignore All Policies**

This option makes any job submitted to this quality of service exempt from all resource usage policies.

### **No Backfill**

This option makes any job submitted to this quality of service exempt from the backfill algorithm.

### **No Reservation**

This option makes any job submitted to this quality of service unable to create a job reservation and, therefore, only able to share resources.

### **Next to Run**

This option makes any job submitted to this quality of service run next. This is accomplished by increasing the start priority of a job, than all of the other queued jobs.

### **Preemptee**

This option makes any job submitted to this quality of service preemptable. A preemptable job can be stopped and requeued if a high priority preemptor job needs to execute.

### **Preemptor**

This option makes any job submitted to this quality of service able to preempt any preemptable job.

### **Provision**

This option makes any job submitted to this quality of service that requests unavailable resources such as an operating system or software to have Workload Manager set up a number of nodes with the correct resources.

### **Reserve Always**

This option makes any job submitted to this quality of service create a reservation. Usually job reservations are created when the job starts but with this option enabled the job will create a reservation immediately.

## Usage Statistics

This is only visible if a profile is being modified.

### **System Utilization**

Shows what percent of the entire system has recently been dedicated to this credential. Each value in this line graph represents the number of dedicated processor seconds used by this credential during the profiling duration divided by the number of dedicated processor seconds in the entire system during the profiling duration. The number of dedicated processor seconds for the entire system for a profiling duration is equal to the size of the profiling duration in seconds multiplied by the number of processors.

# Policies

## **Policies Overview**

Moab Workload Manager has many powerful policies that can be managed to effectively handle as large a workload as possible while satisfying other desires. The policies section is intended to give control over Workload Manager's various policies.

## Backfill

Backfill is an optimization policy that allows a scheduler to make better use of available resources by running jobs out of order. When using Backfill, Workload Manager prioritizes the jobs in the queue into a sorted list with the highest priority job first. Beginning at the top of the list, it starts the jobs one by one until it reaches a job that it cannot start because the necessary resources are not available. Using the start times and wallclock limits of the currently running jobs, Workload Manager then calculates when it will be able to start the job. It reserves that spot (in the future) for the job, and attempts to schedule some of the remaining lower-priority jobs in the gaps left over from the higher-priority jobs. This process continues until Workload Manager has attempted to start all the jobs in the list, until all resources are consumed, or until Workload Manager has considered a specific number of jobs. Backfill allows Workload Manager to achieve a higher utilization than would be otherwise possible, while remaining mostly true to the original job priorities.

Backfill Settings**Policy**

In this field you specify the kind of backfill algorithm Workload Manager uses to schedule jobs.

Backfill Algorithm	Description
<b>FirstFit</b>	Considers jobs in the queue sequentially, beginning with the highest priority and moving down the list.
<b>BestFit</b>	Considers all jobs in the queue and selects the job that best fits the available resources (see the <i>Attribute</i> parameter below).
<b>Greedy</b>	Considers all possible combinations of jobs that can run on the available resources and selects the best combination (see the <i>Attribute</i> parameter below).
<b>Optimistic</b>	If any resource manager reports a state of up, that state will be used.
<b>Preempt</b>	Allows the scheduler to start backfill jobs even if the required walltime is not available.
<b>None</b>	Backfill is not enabled.

**Metric**

This is the criteria used by the backfill algorithm to determine the best jobs to backfill. For example, if *Procs* is selected, a job that requires the exact amount of available processors will be considered the best. This parameter only applies to the *BestFit* and *Greedy* backfill policies.

Metric	Description
<b>Procs</b>	This is the number of processors.
<b>ProcSeconds</b>	This is the number of processors multiplied by the duration of the job, in seconds.
<b>Nodes</b>	This is the number of nodes.
<b>Seconds</b>	This is the duration (or wallclock time) of the job, in seconds.

**Depth**

This is the number of jobs in the queue Workload Manager should consider for backfill. By default, all jobs are considered. If Depth is set, Workload Manager will only consider that number of jobs for backfill scheduling. For example, if there are idle jobs in the queue and Depth is set to 10, only 10 jobs would be considered for backfill. If there are fewer than 10 jobs in the queue, all will be considered. Setting this number higher will result in a higher utilization and better turn-around times, especially for smaller jobs, but may result in low-priority jobs being started before medium-priority jobs. This parameter should be tuned for your specific situation.

**Priority Policy**

- DURATION — The length of the jobs determines which backfill job to preempt.
- HWDURATION
- NONE — No priority policy is used.
- RANDOM

**Chunking Duration**

This field allows an administrator to specify the duration during which freed resources will be aggregated for use by larger jobs.

**Chunk Size**

This field allows an administrator to specify the minimum job size which can utilize chunked resources.

**Virtual Walltime Scaling Factor**

This field allows an administrator to specify the factor by which eligible jobs' wallclock time is virtually scaled.

**Min. Virtual Walltime**

This field allows an administrator to specify the minimum job wallclock time for virtual scaling.

## Fairshare

Fairshare allows the cluster to be shared between different individuals and/or organizations without allowing any individual or organization the ability to monopolize the cluster. This is achieved by tracking how the cluster is used over time by each credential or user, group, class, account, and quality of service (QoS) and by increasing or decreasing the start priorities of jobs waiting to execute. It should be noted that the start priority is used by Workload Manager to decide which jobs get executed first. The higher the start priority, the sooner a job will execute. The information collected about each credential is inserted into what is called a usage window. The amount of time tracked in a window is defined by the system administrator.

System administrators often cannot achieve the cluster sharing they desire without using multiple usage windows. Multiple usage windows allow Workload Manager to balance cluster usage differently by making the most recent window more important than more outdated windows. This is achieved by using

the Usage Window Decay Factor. Essentially, the lower the decay factor, the less important outdated usage windows are.

Fairshare Options

**Interval Length**

This field allows an administrator to define how long each window lasts.

**Depth**

This field allows an administrator to define how many windows should exist.

**Usage Metric**

This field allows an administrator to define how credential usage is tracked.

Metric	Description
<b>Dedicated PES</b>	This field tracks credential usage according to the number of processor equivalent seconds reserved for a job.
<b>Dedicated PS</b>	This field tracks credential usage according to the number of processor seconds reserved for a job.
<b>PDEDICATEDPS</b>	This field tracks dedicated processor seconds scaled by the processor speed of the node.
<b>SDEDICATEDPES</b>	This field tracks dedicated processor-equivalent seconds scaled by the speed of the node.
<b>Utilized PS</b>	This field tracks credential usage according to the number of processor seconds used by a job.

### Decay Factor

This field allows an administrator to define how much of an influence outdated usage windows have in calculating a job's start priority. The lower the Usage Window Decay Factor, the less important the outdated usage windows.

### Fairshare Table

Column	Description
<b>Credential</b>	This field displays the credential type (user, group, account, class, or quality of service (QoS)).
<b>ID</b>	This field displays the credential's unique name.
<b>Policy</b>	This field displays the fairshare credentials policy. Consult the create user, group, class, account, or quality of service (QoS) documentation for more information regarding the policies.
<b>Target</b>	This field displays the fairshare credentials policy. Consult the create user, group, class, account, or quality of service (QoS) documentation for more information regarding the targets.
<b>Usage</b>	This field displays the percentage of the cluster that has been used by this credential ID in comparison to the other credential IDs for this Credential Type.
<b>Current Interval</b>	This field displays the first usage window. The decay factor does not affect this window at all.
<b>Interval 1 - 31</b>	This field displays the usage interval windows 1 through 31. The decay factor affects these windows with the most outdated window being window 31 and the most recent window being window 1.

## Job policies and settings

Contained in this section:

### [Job Limit Settings](#)

This window is used to place system wide restrictions on jobs.

#### **Max. Jobs Started per Iteration**

This field allows an administrator to specify the maximum number of times Workload Manager will attempt to start the job.

#### **Max. Jobs Preempted per Iteration**

This field allows an administrator to define the maximum number of times a job can be preempted by Workload Manager for higher priority jobs.

**Max. Processors per Job**

This field allows an administrator to define the maximum number of processors that can be requested by any jobs.

**Max. Processor-Seconds per Job**

This field allows an administrator to define the maximum number of processor-seconds (total number of procs reserved times the amount of time in seconds that the procs are reserved) that can be requested by any jobs.

**Max. Wallclock**

This field allows an administrator to specify the maximum amount of Wallclock time that can be requested by any single job.

**Wallclock Violation Soft Limit**

This field allows an administrator to define the amount of time Workload Manager allows a job to exceed its wallclock limit before Moab sends mail to the primary administrator.

**Wallclock Violation Limit**

This field allows an administrator to define the amount of time Workload Manager allows a job to exceed its wallclock limit before it is terminated.

**Wallclock Violation Action**

This field allows an administrator to define whether Moab should cancel or preempt a job that has exceeded its wallclock limit.

[Job Defer Settings](#)

**Defer Wait Time**

This field allows an administrator to define the amount of time a job will be held in the deferred state before being released back to the idle job queue.

**Starts Before Defer**

This field allows an administrator to define the amount of time a job will be allowed to fail in its start attempts before being deferred.

**Defers Before Hold**

This field allows an administrator to define the number of times a job can be deferred before it will be placed in batch hold.

**Job Sync Time**

This field allows an administrator to define the length of time after which Workload Manager will change a job's expected state to an unexpected reported state. It should be noted that Workload Manager will not allow a job to run as long as its expected state does not match the state reported by the resource manager.

[Global Job Policy Settings](#)

This window contains job-specific global settings.

**Preempt Policy**

This field allows an administrator to specify how Moab should respond to a preempted job.

Policy	Description
<b>CANCEL</b>	The job is terminated and removed from the queue.
<b>CHECKPOINT</b>	The job is terminated, checkpointed, and left in the queue.
<b>REQUEUE</b>	The job is terminated and left in the queue.
<b>SUSPEND</b>	The active job is suspended.

**Node Match Policy**

This field allows an administrator to specify how compute nodes are selected.

Policy	Description
<b>EXACTNODE</b>	Moab selects as many nodes as requested, regardless of how many tasks it assigns to the same node.
<b>EXACTPROC</b>	Moab selects nodes with exactly the same number of processors configured as are requested per node, even if other nodes have more processors available.
<b>EXACTTASK</b>	Moab selects nodes with exactly the same number of tasks configured as are requested per node, even if other nodes have more tasks available.

**Job Priority Policy**

This field allows an administrator to specify how Moab should track the dynamic aspects of a job's priority.

Policy	Description
<b>ACCRUE</b>	The job will accrue queue time based priority from the time it is submitted unless it violates any of the policies not specified in <code>JOBPRIOEXCEPTIONS</code> .
<b>RESET</b>	The job will accrue priority from the time it is submitted unless it violates any of the <code>JOBPRIOEXCEPTIONS</code> . If the job does violate <code>JOBPRIOEXCEPTIONS</code> , its queue time based priority will be reset to 0.

**Use Machine Speed**

By checking this box, an administrator specifies that a job's wallclock should be increased if the job is executing on a slower node and that a job's wallclock should be decreased if the job is executing on a faster node. The speed of the node is assessed by examining the node speed option located in the list nodes window.

**Enable Multiple Node Jobs**

By checking this box, an administrator specifies that resources may be allocated from multiple nodes to a job.

**Enable Multiple Requirement Jobs**

By checking this box, an administrator specifies that jobs may have more than one requirement.

## Node policies

**Node Task Allocation**

A task is a request for resources that must exist on a single compute node. Each job may have one or more tasks. Workload Manager allocates resources to jobs based on the tasks in the job. This is useful because nodes with multiple processors are usually able to support more than one task at a time. For example, if a job has 2 tasks where each task requires 1 processor and 256 MB of memory, Workload Manager may choose to allocate the job to a dual processor node with 512 MB of memory or to 2 single processor nodes with 256 MB of memory each. The node-task allocation policy determines which tasks may run on the same node.

This is the algorithm Workload Manager uses to determine which tasks may run on the same node.

Policy	Description
<b>Shared optional</b>	Tasks from any job and any user may use available resources on any node.
<b>Shared only</b>	Tasks from any job and any user may use available resources on any node.
<b>Single Job</b>	Only tasks from the same job may run for any given node.
<b>Single Task</b>	Only one task may run on each node.
<b>Single User</b>	For any given node, only tasks from jobs submitted by the same user may run.
<b>Unique User</b>	Any number of tasks from a job may allocate resources from a node if the user has no other jobs running on that node.

## Node Allocation Policy

Node allocation is the process of selecting the best resources, from a list of available resources, to assign to a job. Making this decision intelligently is important in environments with heterogeneous resources or nodes that can support multiple jobs at the same time. This is the algorithm Workload Manager uses to allocate nodes.

Policy	Description
<b>FirstAvailable</b>	Nodes are allocated in the order they are presented by the resource manager.
<b>LastAvailable</b>	Resources are selected so as to minimize the amount of time the resources remain unused after the job completes. This minimizes node-time fragmentation and is useful in systems that have a large number of reservations.
<b>MinResource</b>	Nodes that have the smallest amount of resources that meet the job's requirements are selected.
<b>CPULoad</b>	Nodes that have the maximum amount of available, unused CPU power are selected. This is good for timesharing systems, but is only applicable to jobs starting immediately. For future jobs, the MinResource policy is used.
<b>Contiguous</b>	Nodes are allocated in contiguous (linear) blocks. This is required by the Compaq RMS system.
<b>MaxBalance</b>	Nodes that are as similar as possible to each other are allocated to each job. The most important consideration in determining node similarity is node speed.
<b>Fastest</b>	The fastest available nodes are allocated to each job. Workload Manager determines which nodes are fastest based upon first the node speed and then the processor speed of each node. If neither of these values is available, the nodes are selected randomly.

## Node Availability Policy

Workload Manager will start jobs on nodes that are not full and are not considered busy. Workload Manager considers a node busy according to which Node Availability Policy is set. This is the algorithm Workload Manager uses to determine if a node is busy.

Policy	Description
<b>Combined</b>	Either of the above two conditions is met.
<b>Dedicated</b>	The dedicated (assigned or reserved) resources on the node equal or exceed the configured resources.
<b>Utilized</b>	The utilized (in use) resources on the node equal the configured (total available) resources.

**i** Resources may be dedicated to a user, group, or account for a specific period of time. But some of those resources may not be used during the entire period. This setting allows Workload Manager to differentiate between the two possibilities.

## Partition policies

This section deals with policies relating partitions and their behavior. Below is a list of partition policies.

### Partition Allocation Policy

This is a direct way to assign a peer allocation algorithm when multiple partitions are available for a job. Because clusters are considered partitions, this defines how jobs can be migrated to remote resources if multiple remote clusters can be found. Values and their descriptions are listed in the table below.

Policy	Description
<b>BESTFIT</b>	Allocate resources from the eligible peer with the fewest available resources - measured in tasks (minimizes fragmentation of large resource blocks).
<b>BESTPFIT</b>	Allocate resources from the eligible peer with the fewest available resources - measured in percent of configured resources (minimizes fragmentation of large resource blocks).
<b>FIRSTCOMPLETION</b>	Allocate resources from the eligible peer which can complete the job the soonest (takes into account data staging time and job-specific machine speed).
<b>FIRSTSTART</b>	Allocate resources from the eligible peer which can start the job the soonest.
<b>ROUNDROBIN</b>	Allocate resources from the eligible peer which has been least recently allocated.
<b>LOADBALANCE</b>	Allocate resources from the eligible peer with the most available resources - measured in tasks (balances workload distribution across potential peers).
<b>LOADBALANCEP</b>	Allocate resources from the eligible peer with the most available resources - measured in percent of configured resources (balances workload distribution across potential peers).

## Priority

A job has one start priority that determines when a job will start. The higher the start priority, the sooner a job will start. The job start priority can be anywhere between 1,000,000,000 and -1,000,000,000.

## How to read priorities

A job has one start priority that determines when a job will start. The higher the start priority, the sooner a job starts. Workload Manager uses the priority policies to calculate a job's start priority.

A subcomponent priority of 0 means the subcomponent will be ignored. A positive subcomponent priority means the start priority will be increased. A negative subcomponent priority means the start priority will be decreased. Refer to the documentation below for information about Main Components.

## How to understand the priority window layout

Workload Manager uses 39 components to calculate the start priority. These components are grouped into tabs according to their functionality. The Main Component is different from the subcomponents. Refer to the documentation below for further information about Main Components.

The table shows only idle/queued jobs and their start priority. Only idle/queued jobs are displayed because priority policies do not affect running jobs.

## What are components and subcomponents?

The 7 component groupings are crucial to understanding priorities. The Main Component tab is used only to increase or decrease the subcomponents priorities. The subcomponents increase or decrease the job start priority.

## How the job start priority is calculated

A job has one start priority which is used to decide when a job will start. The higher the start priority the sooner a job will start. Workload Manager uses the priority policies to calculate a job's start priority.

The start priority is calculated by adding all the subcomponents in a group together and multiplying the total of these subcomponents by the Main Component priority. This process is repeated 7 times, or once for each main component. The start priority is a summation of these 7 totals.

It should be noted that if the Main Component priority is set to 0, all of the subcomponent priorities for that Main Component will be ignored.

## How to enable a priority

To enable a priority, two priorities must change. The first priority is the sub component priority and the second priority is the Main Component priority.

For example, to apply a priority of 1 for a user's priority, the user priority in the subcomponent credential's priorities and the Credential Priorities in the Main Component must both be set to 1.

## What does a 0 mean?

If the Main Component priority is set to 0, all of the subcomponent priorities for that Main Component will be ignored. A sub component priority of 0 means the sub component will be ignored.

## Main Priority Components

### **Wait Time Job Service**

This field allows an administrator to increase or decrease all of the Wait Time Job Service priorities. If this is set to 0, all of the subcomponent's priorities for Wait Time Job Service are ignored.

### **Quality of Service Target**

This field allows an administrator to increase or decrease all of the Quality of Service Target's priorities. If this is set to 0, all of the subcomponent's priorities for Quality of Service Target will be ignored.

### **Fairshare Usage**

This field allows an administrator to increase or decrease all of the Fairshare Usage priorities. If this is set to 0, all of the subcomponent's priorities for Fairshare Usage will be ignored.

### **Resource Requests**

This field allows an administrator to increase or decrease all of the Resource Request's priorities. If this is set to 0, all of the subcomponent's priorities for Resource Request's will be ignored.

### **Credential Priorities**

This field allows an administrator to increase or decrease all of the Credential Priority's priorities. If this is set to 0, all of the subcomponent's priorities for Credential Priorities will be ignored.

### **Job Attributes**

This field allows an administrator to increase or decrease all of the Job Attributes priorities. If this is set to 0, all of the subcomponent's priorities for Job Attributes will be ignored.

### **Executing Job Usage**

This field allows an administrator to increase or decrease all of the Executing Job Usage priorities. If this is set to 0, all of the subcomponent's priorities for Executing Job Usage will be ignored. Unlike the other components, this component only effects executing jobs and is only applicable when preemption is used.

## Wait Time Job Services

### **Queue Time**

This field allows an administrator to set the priority of a job according to the minutes the job has waited in the queue.

### **X-Factor (Expansion Factor)**

This field allows an administrator to set the priority of a job according to the expansion factor\* of the job.

### **Policy Violation**

This field allows an administrator to set the priority of a job according to whether the job has violated a usage limit. If the job has violated a usage limit, the job is assigned a policy violation value of 1; otherwise, the job is assigned a policy violation value of 0.

### **By Pass**

This field allows an administrator to set the priority of a job according to the number of other jobs that started execution before this job. The other jobs are only counted if Workload Manager started the other jobs because of a backfill policy.

### QoS Targets

#### **Queue Time**

This field allows an administrator to set the priority of a job according to Quality of Service queue time target. The closer the job is to this target, the higher the Queue Time value.

#### **X-Factor (Expansion Factor)**

This field allows an administrator to set the priority of a job according to Quality of Service expansion factor target. The closer the job is to this target, the higher the Expansion Factor value.

### Credential Priority

#### **User**

This field allows an administrator to set the priority of a job according the user's priority.

#### **Group**

This field allows an administrator to set the priority of a job according the group's priority.

#### **Account**

This field allows an administrator to set the priority of a job according the account's priority.

#### **Class**

This field allows an administrator to set the priority of a job according the class's priority.

#### **QoS**

This field allows an administrator to set the priority of a job according the Quality of Service (QoS)'s priority.

### Job Attribute/State Priority

#### **Attribute**

This field allows an administrator to set the priority of a job according a job's attributes. Refer to the Workload Manager Priority Factors documentation for information on how to set the Job Attributes.

#### **State**

This field allows an administrator to set the priority of a job according a job's state.

### Fairshare Usage Priority

#### **User**

This field allows an administrator to set the user's priority of a job according to the fairshare usage.

**Group**

This field allows an administrator to set the group's priority of a job according to the fairshare usage.

**Account**

This field allows an administrator to set the account's priority of a job according to the fairshare usage.

**Class**

This field allows an administrator to set the class's priority of a job according to the fairshare usage.

**Quality of Service (QoS)**

This field allows an administrator to set the qualities of service (QoS)'s priority of a job according to the fairshare usage.

**Jobs Per User**

This field allows an administrator to set the priority of a job according to the number of jobs currently executing for this user.

**Processor Seconds per User**

This field allows an administrator to set the priority of a job according to the number of processor seconds currently being used by this user.

**Processors per User**

This field allows an administrator to set the priority of a job according to the number of processors currently being used by this user.

**Resource Requests Priority****Node**

This field allows an administrator to set the priority of a job according to the total number of nodes requested by the job. The more nodes requested, the higher the Node value.

**Disk**

This field allows an administrator to set the priority of a job according to the total amount of disk space requested by the job. The more disk space requested, the higher the disk value.

**Memory**

This field allows an administrator to set the priority of a job according to the total amount of memory in megabytes requested by the job. The more memory requested, the higher the Memory value.

**Swap**

This field allows an administrator to set the priority of a job according to the total amount of swap in megabytes requested by the job. The more swap requested, the higher the swap value.

**Processor-Equivalent**

This field allows an administrator to set the priority of a job according to the total number of processor equivalents requested by the job. The more processor equivalents requested, the higher the processor equivalent's value.

**Walltime**

This field allows an administrator to set the priority of a job according to the total amount of walltime seconds requested by the job. The more walltime requested, the higher the Walltime value.

Executing Job Usage Priority**Consumed**

This field allows an administrator to set the priority of a job according to the total number of processor seconds it has consumed. Unlike other components, this component only effects executing jobs and is only applicable when preemption is used.

**Remaining**

This field allows an administrator to set the priority of a job according to the total number of processor seconds it has remaining. Unlike other components this component only effects executing jobs and is only applicable when preemption is used.

**Percentage Consumed**

This field allows an administrator to set the priority of a job according to the percentage of the Wallclock that has been consumed. Unlike other components, this component only affects executing jobs and is only applicable when preemption is used.

Priority Chart Options**Display start priority pie chart**

This option displays a pie chart of the priority components. Negative components are not displayed.

**Display start priority bar graph**

This option displays a bar chart of the priority components. Positive and negative components are displayed.

## Reservation policies

Workload Manager uses reservations to guarantee that a specific amount of resources will be available for a given job or set of users at a particular time. For example, Workload Manager can reserve 20 processors and 10 GB of memory for users Bob and John from Friday 6:00 AM to Saturday 10:00 PM. Workload Manager uses reservations internally to manage backfill, protect job resources, allow service guarantees, support deadlines and QoS, and enable grid scheduling. Workload Manager supports infinite, reoccurring, and one-time reservations.

When backfill is enabled, Workload Manager attempts to schedule lower-priority jobs ahead of a higher-priority job that can't start immediately. In order to ensure that those low-priority jobs don't delay the high-priority job's start time, Workload Manager can reserve the resources needed by the high-priority

job. These are called priority reservations. The reservation policy determines how Workload Manager handles priority reservations.

### Reservation Settings

#### Policy

This is the policy Workload Manager uses when creating priority reservations. These reservations protect the resources a job is using until the job completes.

Policy	Description
<b>CurrentHighest</b>	Existent priority reservations will be relinquished to new jobs with higher priority
<b>Highest</b>	All idle jobs that receive a reservation will keep it until they run even if new jobs are higher priority
<b>Never</b>	No idle jobs receive reservations

#### Depth

This is how many priority reservations Workload Manager will create. A higher value protects the start time of high-priority jobs but may decrease backfill efficiency.

#### Reservations Per Node

This is the maximum number of priority reservations that can be created on any single node. On large SMP systems, this value should be set to approximately twice the number of reservations that exist on the system.

#### Retry Time

This is the period of time Workload Manager attempts to restart a job with a priority reservation that originally failed to start.

#### Creation Policy

This determines which users can create one-time reservations (also called Administrative reservations). This setting is unrelated to priority reservations.

## User access

The User Access Settings page, found by clicking **Policies > User Access...** in the main menu bar or **Policies > User Access** in the navigation pane, allows an administrator to assign roles to each user in the system. These settings are saved in Workload Manager and affect command-line operations, as well as permissions within Cluster Manager. The set of default roles available is:

Role	Description
<b>Admin1</b>	Users with this role are called administrators. They have complete control of Workload Manager.
<b>Admin2</b>	Users with this role are called operators. They have complete control of jobs, nodes, reservations, etc., but cannot modify Workload Manager configuration parameters.
<b>Admin3</b>	Users with this role are called help desk personnel. They can control various aspects of Workload Manager, but cannot modify workload.
<b>Admin4</b>	Users with this role are most likely trusted or experienced users. They have a subset of Admin2 permissions that is different from Admin3's.
<b>Admin5</b>	Users with this role can only view workload and resource information.

## Reporting

### Reporting Overview

Cluster Manager offers a wide assortment of customizable statistics whether they be customized charts, graphs, matrix statistics, quick charts, or reports.

### Custom charts/graphs

This window allows you to create charts and graphs showing statistics over a custom time interval. You can create pie, bar, line, and stacked line charts and, if desired, export those charts as PNG files. To export a chart, right click it and select "save as" from the menu. Zoom in on sections of the line charts by using the mouse to drag a box over the area of interest. The domain and range axes will automatically be adjusted. To return to the original zoomed out view, hold down the left mouse button, drag towards the left, and release the mouse button.

Note that, since charts with large time ranges are sampled for efficiency, the zoomed-in chart may not be as detailed as directly creating a chart for the zoomed-in time range. For example, you might create a line chart for all of 2008 and zoom in on September 1st. The zoomed-in portion of this chart will be less detailed than if you specified the time parameters as 12:00 am Sept 1 to 11:59 pm Sept. 1st when initially creating the chart.

## Credential Based Charts

### Select Credentials

The drop down box allows you to select which credential for which Moab Cluster Manager will calculate statistics or if the cluster-wide statistics should be used. The Display All Credentials option displays all the credentials that have been tracked by the statistics regardless of whether they have any activity recorded. The Display Credentials with Statistics option displays only the credentials that have recorded some type of activity. The Display Listed Credentials option displays only the requested credentials.

**Select Metric**

One can select from the following metrics when creating a chart.

Metric	Description
<b>Backlog Hours****</b>	The total number of processor hours of all jobs waiting in the queue that belong to a specific credential. Processor hours are calculated by multiplying requested processors and wallclock; for example, a job that requests 10 hours of wallclock time and 3 processors has 30 backlog hours while the job is waiting in the queue.
<b>Dedicated Processor Hours</b>	The number of processor hours dedicated to jobs run by this credential, regardless of whether those jobs are actually using those processors.
<b>Executed Jobs</b>	The number of jobs belonging to this credential that have finished execution. If a line graph is selected, each y value is the number of jobs that completed during the profiling interval around the value on the x axis.
<b>Jobs That Met QOS Target**</b>	The percentage of completed jobs that met their QOS target to total jobs.
<b>Queue Hours**</b>	The number of hours a credential's jobs waited in the queue before starting. Note that queue hours are floored to the closest lower integer value. For example, 23.8 would be floored to 23. This also means that if there are only .9 queue hours, then 0 will be reported.
<b>Resource Requests</b>	Causes three fields to be charted - processor-seconds, memory, and wallclock per credentials that either exceeded the requested resource or under-utilized the requested resource.
<b>System Utilization*</b>	This field displays the system utilization percentage and should range between 0% and 100%. Each value is calculated as dedicated processor hours for that profiling interval divided by the total number of processor hours available on that system in that profiling interval (times 100). For example, if the profiling interval was 1 hour, there were 10 total processors, and 5 of those processors were in use for the entire profiling interval, the calculation would be $(5 \text{ proc hours} / 10 \text{ procs} * 1 \text{ hour}) * 100 = 50\%$ . This makes sense intuitively as half the processors were in use during the profiling interval.
<b>Utilized Processor Hours</b>	The number of processor hours utilized by this credential. Processors that are idle yet are dedicated to the credential are not counted. It is recommended that one use dedicated instead of utilized processor hours where possible since utilized processor hours are dependent on the accuracy of the resource manager, and some resource managers are known to report inaccurate utilized processor hours. Dedicated processor hours are calculated based on data from within Moab Workload Manager and are a more dependable statistic.
<b>XFactor** (Expansion Factor)</b>	The expansion factor of a job is defined as $((\text{Queue Time} + \text{Execution Time}) / \text{Requested Wallclock Time})$ .

In addition to the above metrics, one can select any generic metrics that are configured in the system. Generic metric charts are line graphs that show the cumulative generic metric for all nodes dedicated to this credential. For example, if at a certain time user Bob used 3 nodes which each used 10 WATTS, where WATTS was defined as a generic metric, the corresponding point on the line graph for that time would be 30 WATTS.

**i** Because Workload Manager operates by averaging usage across its statistical intervals the values displayed can be misleading. For example, if the statistical interval for Workload Manager was set to 10 minutes for a cluster of 256 processors and one job, which used all 256 processors, was submitted that started and ended in 5 minutes, then an administrator would assume the System Utilization would display 100% of the processors used. However, because the interval was 10 minutes long and the job only ran for 5 minutes, the average System Utilization for the 10 minute interval was 50%.

**i** This field can only be calculated when a job finishes execution.

**i** Requires that Moab is running under a dedicated node model where a node can be running only 1 job at a time.

**i** The accuracy of backlog hour charts is somewhat less precise when a **NODEACCESSPOLICY** is being used.

### Select Calculation Parameters

The pie charts and bar graphs can be customized to display averages, maximums, sums, and/or line graphs, depending on what calculation parameters are appropriate to the usage metric. Some calculation parameters make little sense for certain usage metrics and thus are disabled.

#### Average

The charts will display either the average per job or average per profiling interval, depending on the usage metric. For example, average backlog per profiling interval will show the average amount of backlog in any given profiling interval.

#### Max

The charts will display the maximum value of this usage metric that was achieved in a profiling interval. For example, the maximum System Utilization in a Profiling Interval will show you the highest system utilization level achieved in a single profiling interval.

#### Sum/Total

The charts will display the sum of the usage metric for all profiling intervals in the selected time window. For example, a chart that shows Total Executed Jobs for April 2008 will display the total number of jobs that executed for each of the selected credentials in April of 2008.

#### Line Graph

One can see a line graph showing the values of the usage metric at each profile interval in the requested time frame.

## Chart Type

For formatting, you can choose from Pie Chart, 3D Pie Chart, Bar Graph, 3D Bar Graph, and Line Graph.

## Time Frame

Choose a time frame for the graph. Time frames can be chosen from the basis of Month, Week, Day, Hour, or Custom.

Time Frame	Description
<b>Month</b>	Gathers data from the first of the month to the end of the month. For example, if today was the 13th of October selecting Month would cause a chart to display data from the 1st through the 31st of October.
<b>Week</b>	Gathers data from midnight Sunday morning of the current week to the current time.
<b>Day</b>	Gathers data from the midnight this morning to just before midnight tonight.
<b>Hour</b>	Gathers data starting at the 0th second of the current hour. For example, if the current time was 11:15am selecting Hour would cause a chart to be created with data from 11:00am to the 11:15am.
<b>Custom</b>	Gathers data from the start time and ends at the end time.

## Node Based Charts

### Node Categorization Charts

Creates charts that show node categorization over time. For example, one might create a line chart that will show when nodes were in a hardware failure state or create a bar graph to show how much time nodes spent in user reservations. Node categories are a superset of the standard idle, busy, and down states commonly found in batch systems. Some examples include Hardware Failure, Software Failure, Maintenance, etc. For more detailed information, see the [Moab Workload Manager Administrator's Guide](#).

Creating a stacked line graph brings up a chart window that allows one to dynamically make node categories visible or invisible. It shows these node categories in a tree structure where branches corresponding to node categories can be expanded or contracted. As these node categories are expanded or contracted the chart is immediately updated. For example, one might collapse the hardware failure and software failure categories into the parent down time node category.

### Why do only 4 of the node categories appear on the chart?

By default, Moab will only categorize nodes into the states idle, active, hardwareFailure, and [NONE]/Other. Usually [NONE]/Other means that Moab was not running. In order to see the other node categories, one must set a reservation on the node indicating the desired category. For example, for

node001 to appear as being in the hardware maintenance state one would set a reservation on node001 whose duration matched the length of the hardware maintenance. Administrators wanting accurate node categorization charts should consider using triggers to set reservations when nodes change state.

## Generic Metric Charts

Creates charts that show the generic metric values of nodes over time. For example, if there was a generic metric called WATTS one could create a chart that showed the WATTS value for each in a set of selected nodes. For information on how to configure generic metrics, see the [Moab Workload Manager Administrator's Guide](#).

To create a chart, simply select the desired nodes and generic metric and click the create button. A chart should appear. If there are too many nodes in your system to view in a single chart, the paging buttons will be enabled. Click the Next Page button to see the next 10 nodes.

You can also filter out data sequences which are not of interest. To do this, click the Advanced checkbox at the lower left of the chart. A lower panel should appear allowing you to select nodes that either spike or average above or below a certain threshold. Clicking the Apply Filter button causes this change to be reflected in the chart. For example, one might only choose to see nodes that spike above 80 WATTS. The Aggregate Nodes into One Sequence option allows one to see a single line where each point represents the sum of the values for all selected nodes for the corresponding time. This would allow one to see how the total number of WATTS used by all nodes in the cluster changes over time.

## What to do if you see a warning that generic metrics aren't configured

If a chart cannot be created due to a warning informing you that no generic metrics are configured in Moab Workload Manager, you should check to see that your resource manager is returning generic metric information to Moab Workload Manager. Generic Metrics are usually returned to Moab Workload Manager through the **CLUSTERQUERYURL** configured in your `moab.cfg`.

 The accuracy of generic metric charts is dependent on the resource manager, since this is where Moab gets generic metric information.

## Job Template Charts

This window allows one to create charts that show statistics relating to job templates. To create a chart, simply select the desired job templates, usage metric, calculation parameters, chart type, and time frame and click the create button.

## Why am I warned that no job templates are configured?

If you cannot create a chart due to a warning that statistics for job templates is not configured in Moab Workload Manager, modify your `moab.cfg`. Most likely you either have no job templates configured or you did not add the JSTAT attribute. Moab Workload Manager will not collect statistical data for job templates not configured with a JSTAT attribute. For more information on configuring job templates with JSTAT, see the [Moab Workload Manager Administrators Guide](#).

## Event calendar

The events calendar helps you view the events you are interested in while filtering out those you are not interested in. This can be much more convenient than searching through logs and event files. Once you have finished filtering, you can view the details of the remaining events and correlate this with your nodes' state history.

### Filtering

The first filter you should probably apply is the time filter. After setting a start and end time, you can then select what type of events you wish to see. Use the tree control at the top left of this window to click the event types and/or subtypes you desire. For example, you might choose to see all events, all job events, or job end events only. After making your selection, click the Apply Filter button.

If you are looking for events for a specific object, and you know the object's name, you can filter out all other events by specifying the object type and ID in the fields at the lower left. For example, you might wish to see only events relating to the job **moab.2**. To do this, you would specify an object type of "Job" and an object ID of **moab.2**.

### Viewing Event Details

To view the details of desired events, simply click their colored icon representations in the timeline. Note that on busy schedulers, an icon may represent multiple events. You will be able to see the event type, ID, time, object type, object ID, and any messages from the scheduler pertaining to this event.

### The Node State/Node Category Chart

Below the timeline is a colored line chart. Vertically below any point on the timeline one can see the percentage of nodes that were active, idle, down, etc. This can be useful in determining the events you might want to filter. For example, if you see that 30% of your nodes suddenly went idle, yet you know there was a large backlog during that time, you might want to view all reservation start events. This might help you find an unused reservation responsible for the idle nodes.

## Matrix statistics

Matrix statistics are used both to analyze historic workload and to predict future workload. The left column of the table displays different job processor sizes. The top column displays relative time frames in the format Hours:Minutes:Seconds.

For example, the table below would be understood as follows. For jobs using 1 processor, 12 jobs are completed within the first 15 minutes of execution and 8 jobs are completed after the first 15 minutes and before the first hour of execution. For jobs using 4 processors, 5 jobs are completed within the first 15 minutes of execution and 0 jobs are completed after the first 15 minutes and before the first hour of execution.

Total Completed Jobs	00:15:00	01:00:00
1 Processor	12	8

<b>Total Completed Jobs</b>	<b>00:15:00</b>	<b>01:00:00</b>
<b>4 Processors</b>	5	-

## Matrix Statistics Types

Matrix Statistic Type	Description
<b>Estimated Start Time</b>	This field displays the predicted start time of a submitted job according to the number of processors the job would use. This information can help users determine how many processors they should submit a job to for optimal start time. For example, it may take less time to start a four-hour job submitted to four processors, than to one processor for a sixteen-hour job.
<b>Average X-Factor (Expansion Factor)</b>	This field displays the historic average expansion factor of a job according to the number of processors it used. The expansion factor is calculated using the following equation: $((\text{queue time of a job} + \text{job's duration}) / \text{job duration})$ .
<b>Maximum X-Factor (Expansion Factor)</b>	This field displays the historic maximum job expansion factor of a job according to the number of processors it used. The expansion factor is calculated using the following equation: $((\text{queue time of a job} + \text{job's duration}) / \text{job duration})$ .
<b>Average Job Queue Time</b>	This field displays the historic average wait time before a job starts executing, according to the number of processors it used. Queue time is the number of hours a job waited before it began execution.
<b>Average Job By Pass</b>	This field displays the historic average by pass of a job according to the number of processors it used. By pass is the number of jobs that started execution before this job because of backfill policies. This is useful in recognizing which types of jobs are being by passed by backfill.
<b>Maximum Job By Pass</b>	This field displays the historic maximum by pass of a job according to the number of processors it used. Bypass is the number of jobs that started execution before this job because of backfill policies. This is useful in recognizing which types of jobs are being by passed by backfill.
<b>Total Completed Jobs</b>	This field displays the total number of jobs that completed in the time interval according to the number of processors they used.

Matrix Statistic Type	Description
<b>Cluster Proc Hours Requested (%)</b>	This field displays a breakdown of the requested time on the cluster according to the number of processor hours. Processor hours are the number of processors times the number of hours that they were requested. The sum total of the table is 100%. Each cell inside the table gives the percentage of the total cluster processor hours requested by jobs of that size and duration.
<b>Cluster Proc Hours Utilized (%)</b>	This field displays a breakdown of the utilized time on the cluster according to the number of processor hours. Processor hours are the number of processors times the number of hours that they were utilized. The sum total of the entire table is 100%. Each cell inside the table gives the percentage of the total cluster processor hours utilized by jobs of that size and duration.
<b>Wall Clock Accuracy (%)</b>	This field displays the average Wallclock accuracy, or user estimate accuracy of how long a job would execute, according to the number of processors it used. A value greater than 100 indicates the average user overestimates the job Wallclock time. A value less than 100 indicates the average user underestimates the job wallclock time. A value of 100 indicates the average user estimates the job wallclock accurately.
<b>Total Backfill Count (%)</b>	This field displays the percentage of jobs that were delayed in executing because the backfill policy made them execute later.
<b>Backfill Proc Hours Executed (%)</b>	This field displays the percentage of processor hours for jobs that were delayed in executing because of the backfill policy that later executed.
<b>Job Efficiency (%)</b>	This field displays the average percentage of the CPU that jobs used according to the number of processors of each job.
<b>Quality of Service (QoS) Utilized (%)</b>	This field displays the average percentage of jobs that received their desired quality of service (QoS) according to the number of processors they used.

## Processor usage

This graph displays how the cluster's processors are being used over time. The left bar, or y-axis, displays the number of processors. The bottom bar, or x-axis, displays time. The light yellow color displays the total available processors on the cluster. The dark yellow color displays the processors used

by jobs and job reservations. The blue color displays the processors used by reservations other than job reservations.

The switch statistics option allows for the Available Processors and Jobs Reservations colors to be switched.

## Quick charts/graphs

Quick Charts provides a simple interface for viewing the most common statistics. Statistics are gathered from the first day of the current month to the last day of the current day, week, or month.

### Available Charts

1. System Overview
2. Total Processor Hours Per Account
3. Average Job Queue Time Per Account
4. Total Processor Hours Per User
5. Job Queue Time Per Quality of Service

## Reports

The Reports section (**Reporting > Reports**) allows you to generate a custom Basic, Advanced, or Detailed Summary Report. To create a report, you must use the fields described below.

### Basic Report

#### Select Credentials

The drop down box allows you to select which credential for which Moab Cluster Manager will calculate statistics or if the cluster-wide statistics should be used. The **Display All Credentials** option displays all the credentials that have been tracked by the statistics regardless of whether they have any activity recorded. The **Display Credentials with Statistics** option displays only the credentials that have recorded some type of activity. The **Display Listed Credentials** option displays only the requested credentials.

**Select Usage Metric**

You can select from the following metric when you generate a report:

Metric	Description
<b>Executed Jobs</b>	The number of jobs belonging to this credential that have finished execution. If a line graph is selected, each y value is the number of jobs that completed during the profiling interval around the value on the x axis.
<b>Dedicated Proc. Hours</b>	The number of processor hours dedicated to jobs run by this credential, regardless of whether those jobs are actually using those processors.
<b>Utilized Proc. Hours</b>	The number of processor hours utilized by this credential. Processors that are idle yet are dedicated to the credential are not counted. It is recommended that one use dedicated instead of utilized processor hours where possible since utilized processor hours are dependent on the accuracy of the resource manager, and some resource managers are known to report inaccurate utilized processor hours. Dedicated processor hours are calculated based on data from within Moab Workload Manager and are a more dependable statistic.
<b>System Utilization</b>	This field displays the system utilization percentage and should range between 0% and 100%. Each value is calculated as dedicated processor hours for that profiling interval divided by the total number of processor hours available on that system in that profiling interval (times 100). For example, if the profiling interval was 1 hour, there were 10 total processors, and 5 of those processors were in use for the entire profiling interval, the calculation would be $(5 \text{ proc hours} / 10 \text{ procs} * 1 \text{ hour}) * 100 = 50\%$ . This makes sense intuitively as half the processors were in use during the profiling interval.
<b>Queue Hours</b>	The number of hours a credential's jobs waited in the queue before starting. Note that queue hours are floored to the closest lower integer value. For example, 23.8 would be floored to 23. This also means that if there is only .9 queue hours then 0 will be reported.
<b>BackLog Hours</b>	The number of processor hours of the jobs belonging to this credential that are waiting in the queue.
<b>XFactor (Expansion Factor)</b>	The expansion factor of a job is defined as $((\text{Queue Time} + \text{Execution Time}) / \text{Requested Wallclock Time})$ .
<b>Used Wallclock Hours</b>	The total number of wallclock hours used by this credential.

**Select Calculation Parameters**

Field	Description
<b>Avg Per</b>	<p>If you select this option, you may choose from the following options:</p> <ul style="list-style-type: none"> <li>• Job - the average amount of queue hours, xfactor, or used wallclock hours per job.</li> <li>• Profiling Interval - the average amount of backlog or system utilization in any given profiling interval.</li> </ul>
<b>Max In a Profiling Interval</b>	<p>If you select this option, the report will display the maximum value of this usage metric that was achieved in a profiling interval. For example, the maximum system utilization in a profiling interval will show you the highest system utilization level achieved in a single profiling interval.</p>
<b>Sum/Total</b>	<p>If you select this option, the report will display the total of the selected metric for the selected credential.</p>

## Advanced Report

**Select Credentials**

The drop down box allows you to select which credential for which Moab Cluster Manager will calculate statistics or if the cluster-wide statistics should be used. The **Display All Credentials** option displays all the credentials that have been tracked by the statistics regardless of whether they have any activity recorded. The **Display Credentials with Statistics** option displays only the credentials that have recorded some type of activity. The **Display Listed Credentials** option displays only the requested credentials.

**Categorized By**

The drop-down box allows you to select a credential type by which the report will be categorized. For instance, if you selected User in the first drop-down and Class in the second, you can generate a report that displays all metrics submitted or used by a user under certain classes. The **Display All Credentials** option displays all the credentials that have been tracked by the statistics regardless of whether they have any activity recorded. The **Display Listed Credentials** option displays only the requested credentials.

**Select Usage Metric**

You can select any of the metrics described in the [Basic Report](#) section on this page.

## Detailed Summary Report

**Select Credentials**

The drop-down box allows you to select a credential type on which to run the report. Use the textbox and the Select User popup below the drop-down to select only specific credentials for the report.

Since only one option is available, you do not need to select a usage metric before generating the report.

## Command line charts/graphs/reports

### Command Line Charts & Graphs

You can create many of the pie, bar, and line charts available from the credential tab in the Charts & Graphs window using a command line interface, which can be faster than navigating through windows and menus. The command line interface also facilitates scripting. For example, you could create a cron job that generates charts at regular intervals and puts them in a specific directory.

To create charts from the command line, navigate to the scripts directory. There should be a script entitled `createchart.sh` and a sample properties file called `cmdlinechart.cfg`. If you are running on Windows, you will want to use `createchart.bat` instead of `createchart.sh`. The `cmdlinechart.cfg` file is a properties file that lets you specify information about what timeframe, credentials, and metric you would like to see in your chart. It includes comments explaining what each property does and has example values. The properties file lets you specify how to connect to Moab Workload Manager. You can use this script to connect to a Moab Workload Manager running either locally or remotely over SSH.

If running on a Unix based operating system you would run the following:

```
./createchart.sh cmdlinechart.cfg
```

If running Windows you would use:

```
createchart.bat cmdlinechart.cfg
```

If the scripts do not work, export `MCM_HOME` and use the following command to generate graphs and charts:

```
export MCM_HOME=/home/bob/mcm-7.0.0-5313
java -cp $MCM_HOME/mcm.jar com.moab.commandline.statistics.CreateChart
cmdlinechart.cfg [-- password <PASSWORD>]
```

*Explanation...*

### Command Line Reports

You can also use the command line to create many of the phone bill style reports available from the Reports window. Similar to command line charts, there is a `createreport.sh` script for Unix based operating systems and a `createreport.bat` script for Windows users. The `cmdlinereport.cfg` file is a sample properties file that contains parameters on how to connect to Moab Workload Manager.

To create reports on a Unix system:

```
./createreport.sh cmdlinereport.cfg
```

If running Windows you would run:

```
createreport.bat cmdlinereport.cfg
```

# Diagnostics

## Diagnostics Overview

Diagnostics are intended to give an administrator quick and easy system information for diagnosing potential problems.

## Diagnostics support

The diagnostics support features (**Help > Support Diagnostics...** in the main menu bar) allows the user to run a set of commands that will check the status of various parts of the system. These commands are controlled by a script named, `support.diag.pl`. This should be in your `$PREFIX/tools/moab` directory.

 This window will not work without the `support.diag.pl` script.

The diagnostics support screen allows the user to select from the tree what commands to run using a built in script. The script will then package the output of each command into a file. This file is saved in an output directory specified by the user or by default in `/tmp/`.

If anything should go wrong in the support process, a pop-up box will open saying what happened. If this is not sufficient, please consult the `mcm.log` file for more information.

# Miscellaneous

## Miscellaneous Overview

Various Cluster Manager sections that don't fit in other categories are contained here.

## Console

Cluster Manager communicates directly with Workload Manager. This console displays the commands submitted to Workload Manager from Cluster Manager as well as any information returned by Workload Manager. Workload Manager output messages are highlighted in green while error messages are highlighted in red.

**Prepend path to Moab Workload Manager commands**

This checkbox indicates whether the path the Moab client commands should be prepended to the commands being run. For example, if checked, the command `showq` would become `/opt/moab/bin/showq` (assuming the Moab path was `/opt/moab/bin`). If false, it would stay `showq` and might result in an error if the Moab client commands were not in the default `$PATH` environment variable. This should be checked when running Moab commands and unchecked when running anything else. If the path was prepended for a non-Moab command like `pwd`, it would become `/opt/moab/bin/pwd`, which would result in an error.

**Process Commands**

This field will submit to Workload Manager any text in the Commands text window.

## Debugging and log levels

Allows users to select the log level in Moab Cluster Manager, which can be used to help prepare logs to be accompanied with bug reports. Logs are written to the `<MCM_HOME>/logs/mcm.log` file. Higher logging levels create more detailed logging information, which facilitates debugging but may slow performance.

Below are the logging levels available, listed in order of increasing verbosity.

Level	Name	Description
0	Off	Turns off logging.
1	Fatal	Logs only server events that cause the application to abort.
2	Error	Logs all events that Fatal logs plus error events that might allow the application to continue running.
3	Warn	Logs everything that Error logs plus other minor problems.
4	Info	Logs everything that Warn logs does, plus informational messages that highlight the progress of the application at coarse-grained level.
5	Info- With- Moab- Cmd- Debug	Logs everything that Info logs plus all the interaction with Moab Workload Manager except the frequent (defaults to every 2 seconds) commands to determine if Moab Cluster Manager should refresh data from Moab Workload Manager.
6	Debug	Logs everything that Info-With-Moab-Cmd-Debug level logs plus fine-grained informational events that are useful in debugging.

Level	Name	Description
7	Trace	The most verbose logging level. The only level that logs all interaction with Moab Workload Manager, including the frequent refresh checks ignored by other logging levels.

Users can configure extremely fine grained logging information by editing the `<MCM_HOME>/conf/log4j.properties` file. Using this file, one can set the log level on individual classes or packages within Moab Cluster Manager. It should be noted that configuring individual package or class loggers in the `log4j.properties` file other than the root logger or the logger for `com.moab.api.XMLDebuggingTools` may override the settings applied from this window.

**i** Verbose log levels cause a small performance penalty. Because levels 5 and above log all the interaction with Moab Workload Manager, they can use substantially more memory when connected to larger systems.

## MCM preferences

These preferences control Cluster Manager-specific settings. You can access this window by clicking **Configure > Preferences** in the main menu bar.

### Cache Refresh Rate

The more often Cluster Manager communicates with Workload Manager the more up-to-date the information; however, when Cluster Manager communicates with Workload Manager more often, Workload Manager operates at a slower pace.

#### **Fast**

At this refresh rate Cluster Manager updates its information every minute.

#### **Medium**

At this refresh rate, which is the default refresh rate, Cluster Manager updates its information every 10 minutes.

#### **Slow**

At this refresh rate Cluster Manager updates its information every hour.

### Advanced Settings

Check this box to enable more advanced options throughout Cluster Manager. Currently, the plugin Manager is the only option available.

### Font Size

This option specifies the font size used throughout MCM. Valid sizes are from 9 - 14. Default is 11. MCM must be restarted in order for font size changes to take effect.

## Moab Commands

### **Automatically Process Moab Commands**

If checked, all commands submitted to Moab are automatically processed. If unchecked, all commands are displayed in the commands console for review. Unprocessed commands will have to be manually submitted via the Console.

### **Show Command Output in Pop up Window**

If checked, the output of all commands submitted to will be shown in a pop up window after the commands have run. If not checked, no pop up window will display.

## Navigation

Unchecking the Show Navigation box removes the navigation pane on the left of the summary screen. You can easily replace the pane by checking the box again.

# TORQUE Resource Manager

## Introduction

This section contains some basic introduction information to help you get started using TORQUE. It contains these topics:

- [What is a Resource Manager? on page 2541](#)
- [What are Batch Systems? on page 2541](#)
- [Basic Job Flow on page 2542](#)

### **What is a Resource Manager?**

While TORQUE has a built-in scheduler, `pbs_sched`, it is typically used solely as a *resource manager* with a scheduler making requests to it. Resources managers provide the low-level functionality to start, hold, cancel, and monitor jobs. Without these capabilities, a scheduler alone cannot control jobs.

### **What are Batch Systems?**

While TORQUE is flexible enough to handle scheduling a conference room, it is primarily used in batch systems. Batch systems are a collection of computers and other resources (networks, storage systems, license servers, and so forth) that operate under the notion that the whole is greater than the sum of the parts. Some batch systems consist of just a handful of machines running single-processor jobs, minimally managed by the users themselves. Other systems have thousands and thousands of machines executing users' jobs simultaneously while tracking software licenses and access to hardware equipment and storage systems.

Pooling resources in a batch system typically reduces technical administration of resources while offering a uniform view to users. Once configured properly, batch systems abstract away many of the details involved with running and managing jobs, allowing higher resource utilization. For example, users

typically only need to specify the minimal constraints of a job and do not need to know the individual machine names of each host on which they are running. With this uniform abstracted view, batch systems can execute thousands and thousands of jobs simultaneously.

Batch systems are comprised of four different components: (1) Master Node, (2) Submit/Interactive Nodes, (3) Compute Nodes, and (4) Resources.

Component	Description
<b>Master Node</b>	A batch system will have a master node where <code>pbs_server</code> runs. Depending on the needs of the systems, a master node may be dedicated to this task, or it may fulfill the roles of other components as well.
<b>Submit/Interactive Nodes</b>	Submit or interactive nodes provide an entry point to the system for users to manage their workload. For these nodes, users are able to submit and track their jobs. Additionally, some sites have one or more nodes reserved for interactive use, such as testing and troubleshooting environment problems. These nodes have client commands (such as <code>qsub</code> and <code>qhold</code> ).
<b>Computer Nodes</b>	Compute nodes are the workhorses of the system. Their role is to execute submitted jobs. On each compute node, <code>pbs_mom</code> runs to start, kill, and manage submitted jobs. It communicates with <code>pbs_server</code> on the master node. Depending on the needs of the systems, a compute node may double as the master node (or more).
<b>Resources</b>	Some systems are organized for the express purpose of managing a collection of resources beyond compute nodes. Resources can include high-speed networks, storage systems, license managers, and so forth. Availability of these resources is limited and needs to be managed intelligently to promote fairness and increased utilization.

## Basic Job Flow

The life cycle of a job can be divided into four stages: (1) creation, (2) submission, (3) execution, and (4) finalization.

Stage	Description
<b>Creation</b>	<p>Typically, a submit script is written to hold all of the parameters of a job. These parameters could include how long a job should run (<b>walltime</b>), what resources are necessary to run, and what to execute. The following is an example submit file:</p> <pre>#PBS -N localBlast #PBS -S /bin/sh #PBS -l nodes=1:ppn=2,walltime=240:00:00 #PBS -M user@my.organization.com #PBS -m ea source ~/.bashrc cd \$HOME/work/dir sh myBlast.sh -i -v</pre> <p>This submit script specifies the name of the job (<code>localBlast</code>), what environment to use (<code>/bin/sh</code>), that it needs both processors on a single node (<b>nodes=1:ppn=2</b>), that it will run for at most 10 days, and that TORQUE should email "user@my.organization.com" when the job exits or aborts. Additionally, the user specifies where and what to execute.</p>
<b>Submission</b>	A job is submitted with the <code>qsub</code> command. Once submitted, the policies set by the administration and technical staff of the site dictate the priority of the job and therefore, when it will start executing.
<b>Execution</b>	Jobs often spend most of their lifecycle executing. While a job is running, its status can be queried with <code>qstat</code> .
<b>Finalization</b>	When a job completes, by default, the <code>stdout</code> and <code>stderr</code> files are copied to the directory where the job was submitted.

#### Related topics

- [Overview](#)

## Overview

This section contains some basic information about TORQUE, including how to install and configure it on your system. For details, see these topics:

- [TORQUE installation overview on page 2544](#)
- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)
- [Advanced configuration on page 2558](#)
- [Manual setup of initial server configuration on page 2571](#)
- [Server node file configuration on page 2572](#)
- [Testing server configuration on page 2574](#)

- [TORQUE on NUMA systems on page 2576](#)
- [TORQUE Multi-MOM on page 2580](#)

## TORQUE installation overview

This section contains information about TORQUE architecture and explains how to install TORQUE. It also describes how to install tpackages on compute nodes and how to enable TORQUE as a service.

For details, see these topics:

- [TORQUE architecture on page 2544](#)
- [Installing](#)
- [Compute nodes on page 2549](#)
- [Enabling TORQUE as a service on page 2550](#)

Related topics

- [Troubleshooting on page 2665](#)

## TORQUE architecture

A TORQUE cluster consists of one head node and many compute nodes. The head node runs the `pbs_server` daemon and the compute nodes run the `pbs_mom` daemon. Client commands for submitting and managing jobs can be installed on any host (including hosts not running `pbs_server` or `pbs_mom`).

The head node also runs a scheduler daemon. The scheduler interacts with `pbs_server` to make local policy decisions for resource usage and allocate nodes to jobs. A simple FIFO scheduler, and code to construct more advanced schedulers, is provided in the TORQUE source distribution. Most TORQUE users choose to use a packaged, advanced scheduler such as [Maui](#) or [Moab](#).

Users submit jobs to `pbs_server` using the `qsub` command. When `pbs_server` receives a new job, it informs the scheduler. When the scheduler finds nodes for the job, it sends instructions to run the job with the node list to `pbs_server`. Then, `pbs_server` sends the new job to the first node in the node list and instructs it to launch the job. This node is designated the execution host and is called *Mother Superior*. Other nodes in a job are called *sister MOMs*.

Related topics

- [TORQUE installation overview on page 2544](#)
- [Installing](#)

## Installing TORQUE

These instructions describe how to install and start TORQUE.

## Requirements

### *Supported Operating Systems*

- CentOS 6.5 or later
- Red Hat 6.5 or later
- Scientific Linux 6.5 or later
- SUSE Linux Enterprise Server 11 SP3 or later

**i** CentOS 5.9, Red Hat 5.9 and Scientific Linux 5.9 are supported, largely to continue support for clusters where the compute nodes operating systems cannot be upgraded. We recommend that the TORQUE head node run on the supported operating systems listed above.

### *Software Requirements*

- libxml2-devel package (package name may vary)
- openssl-devel package (package name may vary)
- Tcl/Tk version 8 or later if you plan to build the GUI portion of TORQUE or use a Tcl based scheduler
- If you use [cpuset](#)s, libhwloc 1.1 or later is required (for TORQUE 4.0.0 and later)

If you build TORQUE from source (i.e. clone from github), the following additional software is required:

- gcc
- gcc-c++
- A posix compatible version of make
- libtool 1.5.22
- boost-devel 1.36.0

## Prerequisites

- TORQUE requires certain ports to be open for essential communication:
  - For client communication to `pbs_server`, all privileged ports must be open (ports under 1024).
  - For `pbs_server` communication to `pbs_mom`, the default port is 15003.
  - For `pbs_mom` to `pbs_server`, the default port is 15001.

For more information on how to configure the ports that TORQUE uses for communication, see [Configuring ports on page 2555](#).

**i Important:** If you intend to use TORQUE 5.0.0 with Moab, you must run Moab version 8.0.0 or later. TORQUE 5.0.0 will not work with versions earlier than Moab 8.0.0.

- Make sure your host (with the correct IP address) is in your `/etc/hosts` file.
- The `libxml2-devel`, `openssl-devel`, and `boost-devel` packages must be installed (These packages should already be installed from following the steps in the [Preparing for installation on page 17](#)).

### RHEL 6.5 and CentOS 6.5, and Scientific Linux 6.5:

```
[root]# yum install openssl-devel libtool-devel libxml2-devel boost-devel gcc gcc-c++
```

### SLES

```
[root]# zypper install openssl-devel libtool-devel libxml2-devel boost-devel gcc gcc-c++
```

### RHEL 5 and CentOS 5, and Scientific Linux 5:

```
[root]# yum install openssl-devel libtool-devel libxml2-devel gcc gcc-c++
```

**i Important:** TORQUE requires Boost version 1.36.0 or greater. The `boost-devel` package provided with RHEL 5, CentOS 5, and Scientific Linux 5 is older than this requirement. A new option, `--with-boost-path` has been added to configure (see [Customizing the install on page 2558](#) in the *TORQUE Administrator Guide* for more information). This allows you to point TORQUE to a specific version of boost during make. One way to compile TORQUE without installing Boost is to simply download the Boost version you plan to use from: <http://www.boost.org/users/history/>. Next, untar Boost—you do not need to build it or install it. When you run TORQUE configure, use the `--with-boost-path` option pointed to the extracted Boost directory.

## To install TORQUE

1. Switch the user to root.

```
[user]$ su -
```

2. Download the latest 5.0.0 build from the [Adaptive Computing](#) website. It can also be downloaded via command line.
  - a. Clone the source from github. If you clone the source from github, the `libtool` package must be installed.

**i** If you are using CentOS 5, use these instructions for installing `libtool`:

```
[root]# cd /tmp
[root]# wget http://ftpmirror.gnu.org/libtool/libtool-2.4.2.tar.gz
[root]# tar -xzvf libtool-2.4.2.tar.gz
[root]# cd libtool-2.4.2
[root]# ./configure --prefix=/usr
[root]# make
[root]# make install
[root]# cd /tmp
[root]# git clone https://github.com/adaptivecomputing/torque.git -b
5.0.0 5.0.0
[root]# cd 5.0.0
[root]# ./autogen.sh
```

```
# RHEL 6 and Scientific Linux 6:
[root]# yum install git libtool
```

```
# SLES:
[root]# zypper install libtool
```

```
[root]# git clone https://github.com/adaptivecomputing/torque.git -b 5.0.0 5.0.0
[root]# cd 5.0.0
[root]# ./autogen.sh
```

- b. Get the tarball source distribution.

```
[root]# wget http://www.adaptivecomputing.com/download/torque/torque-
5.0.0.tar.gz -O torque-5.0.0.tar.gz

[root]# tar -xzvf torque-5.0.0.tar.gz
[root]# cd torque-5.0.0/
```

3. Run each of the following commands in order.

```
[root]# ./configure
[root]# make
[root]# make install
```

For information on what options are available to customize the `./configure` command, see [Customizing the install on page 2558](#).

4. Configure the `trqauthd` daemon to start automatically at system boot.

```

* If RHEL distribution, do the following *
[root]# cp contrib/init.d/trqauthd /etc/init.d/
[root]# chkconfig --add trqauthd
[root]# echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf
[root]# ldconfig
[root]# service trqauthd start

* If SLES distribution, do the following *
[root]# cp contrib/init.d/suse.trqauthd /etc/init.d/trqauthd
[root]# chkconfig --add trqauthd
[root]# echo /usr/local/lib > /etc/ld.so.conf.d/torque.conf
[root]# ldconfig
[root]# service trqauthd start

```

5. The `make packages` command can be used to create self-extracting packages that can be copied and executed on your nodes. For information on creating packages and deploying them, see [Compute nodes on page 2549](#).

You will also want to scp the `init.d` scripts to the compute nodes and install them there.

6. Verify that the `/var/spool/torque/server_name` file exists and contains the correct name of the server.

```
[root]# echo <pbs_server's_hostname> > /var/spool/torque/server_name
```

7. By default, TORQUE installs all binary files to `/usr/local/bin` and `/usr/local/sbin`. Make sure the path environment variable includes these directories for both the installation user and the root user.

```
[root]# export PATH=/usr/local/bin:/usr/local/sbin:$PATH
```

8. Initialize `serverdb` by executing the `torque.setup` script.

```
[root]# ./torque.setup root
```

9. Add nodes to the `/var/spool/torque/server_priv/nodes` file. For information on syntax and options for specifying compute nodes, see [Specifying compute nodes on page 2553](#).
10. Configure the MOMs if necessary (see [Configuring TORQUE on compute nodes on page 2554](#) in the TORQUE Administrator Guide).
11. Configure `pbs_server` and `pbs_mom` to start automatically at system boot, and then start their daemons.

```

* If RHEL distribution, do the following *
[root]# cp contrib/init.d/pbs_server contrib/init.d/pbs_mom /etc/init.d
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom
[root]# service pbs_server restart
[root]# service pbs_mom start

* If SLES distribution, do the following *
[root]# cp contrib/init.d/suse.pbs_server /etc/init.d/pbs_server
[root]# cp contrib/init.d/suse.pbs_mom /etc/init.d/pbs_mom
[root]# chkconfig --add pbs_server
[root]# chkconfig --add pbs_mom
[root]# service pbs_server restart
[root]# service pbs_mom start

```

### Related topics

- [Preparing for installation on page 17](#)
- [Installing Moab Workload Manager on page 27](#)
- [Component documentation on page 86](#)

## Compute nodes

Use the Adaptive Computing tpackage system to create self-extracting tarballs which can be distributed and installed on compute nodes. The tpackages are customizable. See the `INSTALL` file for additional options and features.

**i** If you installed TORQUE using the RPMs, you must install and configure your nodes manually by modifying the `/var/spool/torque/mom_priv/config` file of each one. This file is identical for all compute nodes and can be created on the head node and distributed in parallel to all systems.

```

[root]# vi /var/spool/torque/mom_priv/config

$pbsserver      headnode      # hostname running pbs server
$logevent       225           # bitmap of which events to log

[root]# service pbs_mom restart

```

### To create tpackages

1. Configure and make as normal, and then run `make packages`.

```

> make packages
Building ./torque-package-clients-linux-i686.sh ...
Building ./torque-package-mom-linux-i686.sh ...
Building ./torque-package-server-linux-i686.sh ...
Building ./torque-package-gui-linux-i686.sh ...
Building ./torque-package-devel-linux-i686.sh ...
Done.

The package files are self-extracting packages that can be copied and executed on
your production machines. Use --help for options.

```

2. Copy the desired packages to a shared location.

```
> cp torque-package-mom-linux-i686.sh /shared/storage/
> cp torque-package-clients-linux-i686.sh /shared/storage/
```

### 3. Install the tpackages on the compute nodes.

Adaptive Computing recommends that you use a remote shell, such as SSH, to install tpackages on remote systems. Set up shared SSH keys if you do not want to supply a password for each host.

**i** The only required package for the compute node is mom-linux. Additional packages are recommended so you can use client commands and submit jobs from compute nodes.

The following is an example of how to copy and install mom-linux in a distributed fashion.

```
> for i in node01 node02 node03 node04 ; do scp torque-package-mom-linux-i686.sh
${i}:/tmp/. ; done
> for i in node01 node02 node03 node04 ; do scp torque-package-clients-linux-
i686.sh ${i}:/tmp/. ; done
> for i in node01 node02 node03 node04 ; do ssh ${i} /tmp/torque-package-mom-linux-
i686.sh --install ; done
> for i in node01 node02 node03 node04 ; do ssh ${i} /tmp/torque-package-clients-
linux-i686.sh --install ; done
```

Alternatively, you can use a tool like xCAT instead of dsh.

### To use a tool like xCAT

#### 1. Copy the tpackage to the nodes.

```
> prcp torque-package-linux-i686.sh noderange:/destinationdirectory/
```

#### 2. Install the tpackage.

```
> psh noderange /tmp/torque-package-linux-i686.sh --install
```

Although optional, it is possible to use the TORQUE server as a compute node and install a pbs\_mom with the pbs\_server daemon.

### Related topics

- [Installing](#)
- [TORQUE installation overview on page 2544](#)

## Enabling TORQUE as a service

**i** Enabling TORQUE as a service is optional. In order to run TORQUE as a service, you must enable trqauthd. (see [Configuring trqauthd for client commands on page 2556](#)).

The method for enabling TORQUE as a service is dependent on the Linux variant you are using. Startup scripts are provided in the contrib/init.d/ directory of the source package. To enable TORQUE as a service, run the following on the host for the appropriate TORQUE daemon:

- RedHat (as root)

```
> cp contrib/init.d/pbs_mom /etc/init.d/pbs_mom
> chkconfig --add pbs_mom
> cp contrib/init.d/pbs_server /etc/init.d/pbs_server
> chkconfig --add pbs_server
```

- SuSE (as root)

```
> cp contrib/init.d/suse.pbs_mom /etc/init.d/pbs_mom
> insserv -d pbs_mom
> cp contrib/init.d/suse.pbs_server /etc/init.d/pbs_server
> insserv -d pbs_server
```

- Debian (as root)

```
> cp contrib/init.d/debian.pbs_mom /etc/init.d/pbs_mom
> update-rc.d pbs_mom defaults
> cp contrib/init.d/debian.pbs_server /etc/init.d/pbs_server
> update-rc.d pbs_server defaults
```

 You will need to customize these scripts to match your system.

These options can be added to the self-extracting packages. For more details, see the `INSTALL` file.

#### Related topics

- [TORQUE installation overview on page 2544](#)
- [Installing](#)
- [Configuring trqauthd for client commands on page 2556](#)

## Initializing/Configuring TORQUE on the server (pbs\_server)

The TORQUE server (`pbs_server`) contains all the information about a cluster. It knows about all of the MOM nodes in the cluster based on the information in the `$TORQUE_HOME/server_priv/nodes` file (See [Configuring TORQUE on compute nodes on page 2554](#)). It also maintains the status of each MOM node through updates from the MOMs in the cluster (see [pbsnodes on page 2705](#)). All jobs are submitted via [qsub](#) to the server, which maintains a master database of all jobs and their states.

Schedulers such as Moab Workload Manager receive job, queue, and node information from `pbs_server` and submit all jobs to be run to `pbs_server`.

The server configuration is maintained in a file named `serverdb`, located in `$TORQUE_HOME/server_priv`. The `serverdb` file contains all parameters pertaining to the operation of TORQUE plus all of the queues which are in the configuration. For `pbs_server` to run, `serverdb` must be initialized.

You can initialize `serverdb` in two different ways, but the recommended way is to use the `./torque.setup` script:

- As root, execute `./torque.setup` from the build directory (see [./torque.setup](#) on page 2552).
- Use `pbs_server -t create` (see [Initializing/Configuring TORQUE on the server \(pbs\\_server\)](#) on page 2551).

Restart `pbs_server` after initializing `serverdb`.

```
> qterm
> pbs_server
```

## ./torque.setup

The `torque.setup` script uses `pbs_server -t create` to initialize `serverdb` and then adds a user as a manager and operator of TORQUE and other commonly used attributes. The syntax is as follows:

`/torque.setup username`

```
> ./torque.setup ken
> qmgr -c 'p s'

#
# Create queues and set their attributes.
#
#
# Create and define queue batch
#
create queue batch
set queue batch queue_type = Execution
set queue batch resources_default.nodes = 1
set queue batch resources_default.walltime = 01:00:00
set queue batch enabled = True
set queue batch started = True
#
# Set server attributes.
#
set server scheduling = True
set server acl_hosts = kmn
set server managers = ken@kmn
set server operators = ken@kmn
set server default_queue = batch
set server log_events = 511
set server mail_from = adm
set server node_check_rate = 150
set server tcp_timeout = 6
set server mom_job_sync = True
set server keep_completed = 300
```

## pbs\_server -t create

The `-t create` option instructs `pbs_server` to create the `serverdb` file and initialize it with a minimum configuration to run `pbs_server`.

```
> pbs_server -t create
```

To see the configuration and verify that TORQUE is configured correctly, use [qmgr](#):

```

> qmgr -c 'p s'
#
# Set server attributes.
#
set server acl_hosts = kmn
set server log_events = 511
set server mail_from = adm
set server node_check_rate = 150
set server tcp_timeout = 6

```

A single queue named batch and a few needed server attributes are created.

This section contains these topics:

- [Specifying compute nodes on page 2553](#)
- [Configuring TORQUE on compute nodes on page 2554](#)
- [Finalizing configurations on page 2558](#)

Related topics

- [Appendix C: Node manager \(MOM\) configuration on page 2783](#)
- [Advanced configuration on page 2558](#)

## Specifying compute nodes

The environment variable `TORQUE_HOME` is where configuration files are stored. If you used the default locations during installation, you do not need to specify the `TORQUE_HOME` environment variable.

The `pbs_server` must recognize which systems on the network are its compute nodes. Specify each node on a line in the server's nodes file. This file is located at `TORQUE_HOME/server_priv/nodes`. In most cases, it is sufficient to specify just the names of the nodes on individual lines; however, various properties can be applied to each node.

 Only a root user can access the `server_priv` directory.

Syntax of nodes file:

```
node-name[:ts] [np=] [gpus=] [properties]
```

- The **node-name** must match the hostname on the node itself, including whether it is fully qualified or shortened.
- The **[:ts]** option marks the node as timeshared. Timeshared nodes are listed by the server in the node status report, but the server does not allocate jobs to them.
- The **[np=]** option specifies the number of virtual processors for a given node. The value can be less than, equal to, or greater than the number of physical processors on any given node.
- The **[gpus=]** option specifies the number of GPUs for a given node. The value can be less than, equal to, or greater than the number of physical GPUs on any given node.

- The node processor count can be automatically detected by the TORQUE server if **auto\_node\_np** is set to TRUE. This can be set using this command:

```
qmgr -c set server auto_node_np = True
```

Setting **auto\_node\_np** to TRUE overwrites the value of **np** set in `TORQUE_HOME/server_priv/nodes`.

- The **[properties]** option allows you to specify arbitrary strings to identify the node. Property strings are alphanumeric characters only and must begin with an alphabetic character.
- Comment lines are allowed in the nodes file if the first non-white space character is the pound sign (#).

The following example shows a possible node file listing.

`TORQUE_HOME/server_priv/nodes`:

```
# Nodes 001 and 003-005 are cluster nodes
#
node001 np=2 cluster01 rackNumber22
#
# node002 will be replaced soon
node002:ts waitingToBeReplaced
# node002 will be replaced soon
#
node003 np=4 cluster01 rackNumber24
node004 cluster01 rackNumber25
node005 np=2 cluster01 rackNumber26 RAM16GB
node006
node007 np=2
node008:ts np=4
...
```

### Related topics

- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)

## Configuring TORQUE on compute nodes

If using TORQUE self-extracting packages with default compute node configuration, no additional steps are required and you can skip this section.

If installing manually, or advanced compute node configuration is needed, edit the `TORQUE_HOME/mom_priv/config` file on each node. The recommended settings follow.

`TORQUE_HOME/mom_priv/config`:

```
$pbsserver      headnode      # hostname running pbs server
$logevent       225              # bitmap of which events to log
```

This file is identical for all compute nodes and can be created on the head node and distributed in parallel to all systems.

### Related topics

- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)

## Configuring ports

You can optionally configure the various ports that TORQUE uses for communication. Most ports can be configured multiple ways. The ports you can configure are:

- [pbs\\_server listening port](#)
- [pbs\\_mom listening port](#)
- [port pbs\\_server uses to communicate to the pbs\\_mom](#)
- [port pbs\\_mom uses to communicate to the pbs\\_server](#)
- [port client commands use to communicate to the pbs\\_server](#)
- [port trqauthd uses to communicate to the pbs\\_server](#)



If you are running pbspro on the same system, be aware that it uses the same environment variables and `/etc/services` entries.

### Configuring the pbs\_server listening port

To configure the port the `pbs_server` listens on, follow any of these steps:

- Set an environment variable called `PBS_BATCH_SERVICE_PORT` to the port desired.
- Edit the `/etc/services` file and set `pbs port_num/tcp`.
- Start `pbs_server` with the `-p` option.

```
$ pbs_server -p port_num
```

- Edit the `$PBS_HOME/server_name` file and change `server_name` to `server_name:<port_num>`
- Start `pbs_server` with the `-H` option.

```
$ pbs_server -H server_name:port_num
```

### Configuring the pbs\_mom listening port

To configure the port the `pbs_mom` listens on, follow any of these steps:

- Set an environment variable called `PBS_MOM_SERVICE_PORT` to the port desired.
- Edit the `/etc/services` file and set `pbs_mom port_num/tcp`.
- Start `pbs_mom` with the `-M` option.

```
$ pbs_mom -M port_num
```

- Edit the `nodes` file entry for that list: add `mom_service_port=port_num`.

### Configuring the port pbs\_server uses to communicate with pbs\_mom

To configure the port the `pbs_server` uses to communicate with `pbs_mom`, follow any of these steps:

- Set an environment variable called `PBS_MOM_SERVICE_PORT` to the port desired.
- Edit the `/etc/services` file and set `pbs_mom port_num/tcp`.
- Start `pbs_mom` with the `-M` option.

```
$ pbs_server -M port_num
```

## Configuring the port `pbs_mom` uses to communicate with `pbs_server`

To configure the port the `pbs_mom` uses to communicate with `pbs_server`, follow any of these steps:

- Set an environment variable called `PBS_BATCH_SERVICE_PORT` to the port desired.
- Edit the `/etc/services` file and set `pbs port_num/tcp`.
- Start `pbs_mom` with the `-S` option.

```
$ pbs_mom -p port_num
```

- Edit the `nodes` file entry for that list: add `mom_service_port=port_num`.

## Configuring the port client commands use to communicate with `pbs_server`

To configure the port client commands use to communicate with `pbs_server`, follow any of these steps:

- Edit the `/etc/services` file and set `pbs port_num/tcp`.
- Edit the `$PBS_HOME/server_name` file and change `server_name` to `server_name:<port_num>`

## Configuring the port `trqauthd` uses to communicate with `pbs_server`

To configure the port `trqauthd` uses to communicate with `pbs_server`, follow any of these steps:

- Edit the `$PBS_HOME/server_name` file and change `server_name` to `server_name:<port_num>`

### Related topics

- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)
- [pbs\\_server](#)
- [pbs\\_mom](#)
- [trqauthd](#)
- [client commands](#)

## Configuring `trqauthd` for client commands

`trqauthd` is a daemon used by TORQUE client utilities to authorize user connections to `pbs_server`. Once started, it remains resident. TORQUE client utilities then communicate with `trqauthd` on port 15005 on the loopback interface. It is multi-threaded and can handle large volumes of simultaneous requests.

## Running trqauthd

trqauthd must be run as root. It must also be running on any host where TORQUE client commands will execute.

By default, trqauthd is installed to `/usr/local/bin`.

trqauthd can be invoked directly from the command line or by the use of init.d scripts which are located in the `contrib/init.d` directory of the TORQUE source.

There are three `init.d` scripts for trqauthd in the `contrib/init.d` directory of the TORQUE source tree:

Script	Description
<b>debian.trqauthd</b>	Used for apt-based systems (debian, ubuntu are the most common variations of this)
<b>suse.trqauthd</b>	Used for suse-based systems
<b>trqauthd</b>	An example for other package managers (Redhat, Scientific, CentOS, and Fedora are some common examples)

**i** You should edit these scripts to be sure they will work for your site.

Inside each of the scripts are the variables `PBS_DAEMON` and `PBS_HOME`. These two variables should be updated to match your TORQUE installation. `PBS_DAEMON` needs to point to the location of trqauthd. `PBS_HOME` needs to match your TORQUE installation.

Choose the script that matches your dist system and copy it to `/etc/init.d`. If needed, rename it to **trqauthd**.

### To start the daemon

```
/etc/init.d/trqauthd start
```

### To stop the daemon

```
/etc/init.d/trqauthd stop
```

OR

```
service trqauthd start/stop
```

**i** If you receive an error that says "Could not open socket in trq\_simple\_connect. error 97" and you use a CentOS, RedHat, or Scientific Linux 6+ operating system, check your `/etc/hosts` file for multiple entries of a single host name pointing to the same IP address. Delete the duplicate(s), save the file, and launch trqauthd again.

## Related topics

- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)

## Finalizing configurations

After configuring the `serverdb` and the `server_priv/nodes` files, and after ensuring minimal MOM configuration, restart the `pbs_server` on the server node and the `pbs_mom` on the compute nodes.

Compute Nodes:

```
> pbs_mom
```

Server Node:

```
> qterm -t quick
> pbs_server
```

After waiting several seconds, the `pbsnodes -a` command should list all nodes in state `free`.

## Related topics

- [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)

## Advanced configuration

This section contains information about how you can customize the installation and configure the server to ensure that the server and nodes are communicating correctly. For details, see these topics:

- [Customizing the install on page 2558](#)
- [Server configuration on page 2566](#)

## Related topics

- [Appendix B: Server parameters on page 2766](#)

## Customizing the install

The TORQUE `configure` command has several options available. Listed below are some suggested options to use when running `./configure`.

- By default, TORQUE does not install the admin manuals. To enable this, use `--enable-docs`.
- By default, only children MOM processes use syslog. To enable syslog for all of TORQUE, use `--enable-syslog`.

Table 4-3: Optional Features

Option	Description
<b>--disable-clients</b>	Directs TORQUE not to build and install the TORQUE client utilities such as qsub, qstat, qdel, etc.
<b>--disable-FEATURE</b>	Do not include FEATURE (same as --enable-FEATURE=no).
<b>--disable-lib-tool-lock</b>	Avoid locking (might break parallel builds).
<b>--disable-mom</b>	Do not include the MOM daemon.
<b>--disable-mom-check-spool</b>	Don't check free space on spool directory and set an error.
<b>--disable-posixmemlock</b>	Disable the MOM's use of mlockall. Some versions of OSs seem to have buggy POSIX MEMLOCK.
<b>--disable-priv-ports</b>	Disable the use of privileged ports for authentication. Some versions of OSX have a buggy bind () and cannot bind to privileged ports.
<b>--disable-qsub-keep-override</b>	Do not allow the qsub -k flag to override -o -e.
<b>--disable-server</b>	Do not include server and scheduler.
<b>--disable-shell-pipe</b>	Give the job script file as standard input to the shell instead of passing its name via a pipe.
<b>--disable-spool</b>	If disabled, TORQUE will create output and error files directly in \$HOME/.pbs_spool if it exists or in \$HOME otherwise. By default, TORQUE will spool files in TORQUE_HOME/spool and copy them to the users home directory when the job completes.
<b>--disable-xopen-networking</b>	With HPUX and GCC, don't force usage of XOPEN and libxnet.
<b>--enable-acct-x</b>	Enable adding x attributes to accounting log.

Option	Description
<b>--enable-array</b>	Setting this under IRIX enables the SGI Origin 2000 parallel support. Normally autodetected from the <code>/etc/config/array</code> file.
<b>--enable-blcr</b>	Enable BLCR support.
<b>--enable-cpa</b>	Enable Cray's CPA support.
<b>--enable-cpu-set</b>	Enable Linux 2.6 kernel cpusets. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  It is recommended that you turn on this feature to prevent a job from expanding across more CPU cores than it is assigned. </div>
<b>--enable-debug</b>	Prints debug information to the console for <code>pbs_server</code> and <code>pbs_mom</code> while they are running. (This is different than <b>--with-debug</b> which will compile with debugging symbols.)
<b>--enable-dependency-tracking</b>	Do not reject slow dependency extractors.
<b>--enable-fast-install[=PKGS]</b>	Optimize for fast installation [default=yes].
<b>--enable-FEATURE [ARG=yes]</b>	Include FEATURE [ARG=yes].
<b>--enable-file-sync</b>	Open files with sync on each write operation. This has a negative impact on TORQUE performance. This is disabled by default.
<b>--enable-force-nodfile</b>	Forces creation of nodfile regardless of job submission parameters. Not on by default.
<b>--enable-gcc-warnings</b>	Enable gcc strictness and warnings. If using gcc, default is to error on any warning.
<b>--enable-geometry-requests</b>	TORQUE is compiled to use <a href="#">procs_bitmap</a> during job submission.
<b>--enable-gui</b>	Include the GUI-clients.

Option	Description
<b>--enable-main-tainer-mode</b>	This is for the autoconf utility and tells autoconf to enable so called rebuild rules. See <a href="#">maintainer mode</a> for more information.
<b>--enable-maxdefault</b>	<p>Turn on the RESOURCEMAXDEFAULT flag.</p> <div style="border: 1px solid black; padding: 5px;"> <p><b>i</b> Versions of TORQUE earlier than 2.4.5 attempted to apply queue and server defaults to a job that didn't have defaults specified. If a setting still did not have a value after that, TORQUE applied the queue and server maximum values to a job (meaning, the maximum values for an applicable setting were applied to jobs that had no specified or default value).</p> <p>In TORQUE 2.4.5 and later, the queue and server maximum values are no longer used as a value for missing settings. To re-enable this behavior in TORQUE 2.4.5 and later, use <code>--enable-maxdefault</code>.</p> </div>
<b>--enable-nochildsignal</b>	Turn on the NO_SIGCHLD flag.
<b>--enable-nodemask</b>	Enable nodemask-based scheduling on the Origin 2000.
<b>--enable-pemask</b>	Enable pemask-based scheduling on the Cray T3e.
<b>--enable-plock-daemons[=ARG]</b>	Enable daemons to lock themselves into memory: logical-or of 1 for pbs_server, 2 for pbs_scheduler, 4 for pbs_mom (no argument means 7 for all three).
<b>--enable-quick-commit</b>	Turn on the QUICKCOMMIT flag.
<b>--enable-shared[=PKGS]</b>	Build shared libraries [default=yes].
<b>--enable-shell-use-argv</b>	Enable this to put the job script name on the command line that invokes the shell. Not on by default. Ignores --enable-shell-pipe setting.
<b>--enable-sp2</b>	Build PBS for an IBM SP2.
<b>--enable-srfs</b>	Enable support for SRFS on Cray.

Option	Description
<b>--enable-static [=PKGS]</b>	Build static libraries [default=yes].
<b>--enable-syslog</b>	Enable (default) the use of syslog for error reporting.
<b>--enable-tcl-qstat</b>	Setting this builds qstat with Tcl interpreter features. This is enabled if Tcl is enabled.
<b>--enable-unix-sockets</b>	Enable the use of Unix Domain sockets for authentication.

Table 4-4: Optional packages

Option	Description
<b>--with-blcr=DIR</b>	BLCR installation prefix (Available in versions 2.5.6 and 3.0.2 and later).
<b>--with-blcr-include=DIR</b>	Include path for libcr.h (Available in versions 2.5.6 and 3.0.2 and later).
<b>--with-blcr-lib=DIR</b>	Lib path for libcr (Available in versions 2.5.6 and 3.0.2 and later).
<b>--with-blcr-bin=DIR</b>	Bin path for BLCR utilities (Available in versions 2.5.6 and 3.0.2 and later).

Option	Description
<b>--with-boost-path=DIR</b>	<p>Set the path to the Boost header files to be used during make. This option does not require Boost to be built or installed.</p> <p>The <code>--with-boost-path</code> value must be a directory containing a sub-directory called <code>boost</code> that contains the <code>boost.hpp</code> files.</p> <p>For example, if downloading the boost 1.55.0 source tarball to the adaptive user's home directory:</p> <pre data-bbox="586 579 1433 800" style="border: 1px dashed gray; padding: 5px;"> [adaptive]\$ cd ~ [adaptive]\$ wget http://sourceforge.net/projects/boost/files/boost/1.55.0/boost_1_55_0.tar.gz/download [adaptive]\$ tar xzf boost_1_55_0.tar.gz [adaptive]\$ ls boost_1_55_0 boost boost-build.jam ...</pre> <p>In this case use <code>--with-boost-path=/home/adaptive/boost_1_55_0</code> during configure.</p> <p>Another example would be to use an installed version of Boost. If the installed Boost header files exist in <code>/usr/include/boost/*.hpp</code>, use <code>--with-boost-path=/usr/include</code>.</p>
<b>--with-cpa-include=DIR</b>	Include path for <code>cpalib.h</code> .
<b>--with-cpa-lib=DIR</b>	Lib path for <code>libcpalib</code> .
<b>--with-debug=no</b>	Do not compile with debugging symbols.
<b>--with-default-server-r=HOSTNAME</b>	Set the name of the computer that clients will access when no machine name is specified as part of the queue name. It defaults to the hostname of the machine on which PBS is being compiled.
<b>--with-environ=PATH</b>	Set the path containing the environment variables for the daemons. For SP2 and AIX systems, suggested setting is to <code>/etc/environment</code> . Defaults to the file <code>"pbs_environment"</code> in the <code>server-home</code> . Relative paths are interpreted within the context of the <code>server-home</code> .
<b>--with-gnu-ld</b>	Assume the C compiler uses GNU ld [default=no].
<b>--with-mail-domain=MAILDOMAIN</b>	Override the default domain for outgoing mail messages, i.e. <code>"user@maildomain"</code> . The default maildomain is the hostname where the job was submitted from.

Option	Description
<b>--with-modulefiles[=DIR]</b>	Use module files in specified directory [/etc/modulefiles].
<b>--with-momlogdir</b>	Use this directory for MOM logs.
<b>--with-momlogsuffix</b>	Use this suffix for MOM logs.
<b>--without-PACKAGE</b>	Do not use PACKAGE (same as --with-PACKAGE=no).
<b>--without-readline</b>	Do not include readline support (default: included if found).
<b>--with-PACKAGE[=ARG]</b>	Use PACKAGE [ARG=yes].
<b>--with-pam=DIR</b>	Directory that holds the system PAM modules. Defaults to /lib(64)/security on Linux.
<b>--with-pic</b>	Try to use only PIC/non-PIC objects [default=use both].
<b>--with-qstatrc-file=FILE</b>	Set the name of the file that qstat will use if there is no ".qstatrc" file in the directory where it is being invoked. Relative path names will be evaluated relative to the server home directory (see above). If this option is not specified, the default name for this file will be set to "qstatrc" (no dot) in the server home directory.
<b>--with-rcp</b>	One of "scp", "rcp", "mom_rcp", or the full path of a remote file copy program. scp is the default if found, otherwise mom_rcp is used. Some rcp programs don't always exit with valid error codes in case of failure. mom_rcp is a copy of BSD rcp included with this source that has correct error codes, but it is also old, unmaintained, and doesn't have large file support.
<b>--with-sched=TYPE</b>	Sets the scheduler type. If TYPE is "c", the scheduler will be written in C. If TYPE is "tcl" the server will use a Tcl based scheduler. If TYPE is "basl", TORQUE will use the rule based scheduler. If TYPE is "no", then no scheduling is done. "c" is the default.
<b>--with-sched-code=PATH</b>	Sets the name of the scheduler to use. This only applies to BASL schedulers and those written in the C language. For C schedulers this should be a directory name and for BASL schedulers a filename ending in ".basl". It will be interpreted relative to srctree/src/schedulers.SCHD_TYPE/samples. As an example, an appropriate BASL scheduler relative path would be "nas.basl". The default scheduler code for "C" schedulers is "fifo".

Option	Description
<b>--with-scp</b>	In TORQUE 2.1 and later, SCP is the default remote copy protocol. See <a href="#">--with-rcp</a> if a different protocol is desired.
<b>--with-sendmail[=FILE]</b>	Sendmail executable to use.
<b>--with-server-home=DIR</b>	Set the server home/spool directory for PBS use. Defaults to /var/spool/torque.
<b>--with-server-name-file=FILE</b>	Set the file that will contain the name of the default server for clients to use. If this is not an absolute pathname, it will be evaluated relative to the server home directory that either defaults to /usr/spool/PBS or is set using the --with-server-home option to configure. If this option is not specified, the default name for this file will be set to "server_name".
<b>--with-tcl</b>	Directory containing tcl configuration (tclConfig.sh).
<b>--with-tclatrsep=CHAR</b>	Set the Tcl attribute separator character this will default to "." if unspecified.
<b>--with-tclinclude</b>	Directory containing the public Tcl header files.
<b>--with-tclx</b>	Directory containing tclx configuration (tclxConfig.sh).
<b>--with-tk</b>	Directory containing tk configuration (tkConfig.sh).
<b>--with-tkinclude</b>	Directory containing the public Tk header files.
<b>--with-tkx</b>	Directory containing tkx configuration (tkxConfig.sh).
<b>--with-tmpdir=DIR</b>	Set the tmp directory that pbs_mom will use. Defaults to "/tmp". This is a Cray-specific feature.
<b>--with-xauth=PATH</b>	Specify path to xauth program.

## HAVE\_WORDEXP

`Wordxp()` performs a shell-like expansion, including environment variables. By default, `HAVE_WORDEXP` is set to **1** in `src/pbs_config.h`. If set to **1**, will limit the characters that can be used in a job name to those allowed for a file in the current environment, such as BASH. If set to **0**, any valid character for the file system can be used.

If a user would like to disable this feature by setting `HAVE_WORDEXP` to **0** in `src/include/pbs_config.h`, it is important to note that the error and the output file names will not expand environment

variables, including `$PBS_JOBID`. The other important consideration is that characters that BASH dislikes, such as `0`, will not be allowed in the output and error file names for jobs by default.

#### Related topics

- [Advanced configuration on page 2558](#)
- [Server configuration on page 2566](#)

## Server configuration

See these topics for details:

- [Server configuration overview on page 2566](#)
- [Name service configuration on page 2566](#)
- [Configuring job submission hosts on page 2566](#)
- [Configuring TORQUE on a multi-homed server on page 2568](#)
- [Architecture specific notes on page 2568](#)
- [Specifying non-root administrators on page 2568](#)
- [Setting up email on page 2568](#)
- [Using MUNGE authentication on page 2569](#)
- [Setting up the MOM hierarchy on page 2570](#)

## Server configuration overview

There are several steps to ensure that the server and the nodes are completely aware of each other and able to communicate directly. Some of this configuration takes place within TORQUE directly using the `qmgr` command. Other configuration settings are managed using the `pbs_server` nodes file, DNS files such as `/etc/hosts` and the `/etc/hosts.equiv` file.

## Name service configuration

Each node, as well as the server, must be able to resolve the name of every node with which it will interact. This can be accomplished using `/etc/hosts`, DNS, NIS, or other mechanisms. In the case of `/etc/hosts`, the file can be shared across systems in most cases.

A simple method of checking proper name service configuration is to verify that the server and the nodes can "ping" each other.

## Configuring job submission hosts

### Using RCmd authentication

When jobs can be submitted from several different hosts, these hosts should be trusted via the R\* commands (such as `rsh` and `rcp`). This can be enabled by adding the hosts to the `/etc/hosts.equiv` file of the machine executing the `pbs_server` daemon or using other R\* command authorization methods. The

exact specification can vary from OS to OS (see the man page for **ruserok** to find out how your OS validates remote users). In most cases, configuring this file is as simple as adding a line to your `/etc/hosts.equiv` file, as in the following:

`/etc/hosts.equiv:`

```
#[+ | -] [hostname] [username]
mynode.myorganization.com
.....
```

Either of the hostname or username fields may be replaced with a wildcard symbol (+). The (+) may be used as a stand-alone wildcard but not connected to a username or hostname, e.g., `+node01` or `+user01`. However, a (-) may be used in that manner to specifically exclude a user.



Following the Linux man page instructions for `hosts.equiv` may result in a failure. You cannot precede the user or hostname with a (+). To clarify, `node1 +user1` will not work and **user1** will not be able to submit jobs.

For example, the following lines will not work or will not have the desired effect:

```
+node02 user1
node02 +user1
```

These lines will work:

```
node03 +
+ jsmith
node04 -tjones
```

The most restrictive rules must precede more permissive rules. For example, to restrict user `tsmith` but allow all others, follow this format:

```
node01 -tsmith
node01 +
```

Please note that when a hostname is specified, it must be the fully qualified domain name (FQDN) of the host. Job submission can be further secured using the server or queue **acl\_hosts** and **acl\_host\_enabled** parameters (for details, see [Queue attributes on page 2626](#)).

### Using the "submit\_hosts" service parameter

Trusted submit host access may be directly specified without using RCmd authentication by setting the server [submit\\_hosts](#) parameter via **qmgr** as in the following example:

```
> qmgr -c 'set server submit_hosts = host1'
> qmgr -c 'set server submit_hosts += host2'
> qmgr -c 'set server submit_hosts += host3'
```



Use of **submit\_hosts** is potentially subject to DNS spoofing and should not be used outside of controlled and trusted environments.

## Allowing job submission from compute hosts

If preferred, all compute nodes can be enabled as job submit hosts without setting `.rhosts` or `hosts.equiv` by setting the [allow\\_node\\_submit](#) parameter to **true**.

## Configuring TORQUE on a multi-homed server

If the `pbs_server` daemon is to be run on a multi-homed host (a host possessing multiple network interfaces), the interface to be used can be explicitly set using the [SERVERHOST](#) parameter.

## Architecture specific notes

With some versions of Mac OS/X, it is required to add the line `$restricted *.<DOMAIN>` to the `pbs_mom` configuration file. This is required to work around some socket bind bugs in the OS.

## Specifying non-root administrators

By default, only root is allowed to start, configure and manage the `pbs_server` daemon. Additional trusted users can be authorized using the parameters **managers** and **operators**. To configure these parameters use the [qmgr](#) command, as in the following example:

```
> qmgr
Qmgr: set server managers += josh@*.fsc.com
Qmgr: set server operators += josh@*.fsc.com
```

All manager and operator specifications must include a user name and either a fully qualified domain name or a host expression.

**i** To enable all users to be trusted as both operators and administrators, place the + (plus) character on its own line in the `server_priv/acl_svr/operators` and `server_priv/acl_svr/managers` files.

## Setting up email

Moab relies on emails from TORQUE about job events. To set up email, do the following:

### To set up email

1. Use the `--with-sendmail` configure option at configure time. TORQUE needs to know where the email application is. If this option is not used, TORQUE tries to find the `sendmail` executable. If it isn't found, TORQUE cannot send emails.

```
> ./configure --with-sendmail=<path_to_executable>
```

2. Set `mail_domain` in your server settings. If your domain is `clusterresources.com`, execute:

```
> qmgr -c 'set server mail_domain=clusterresources.com'
```

3. (Optional) You can override the default [mail\\_body\\_fmt](#) and [mail\\_subject\\_fmt](#) values via [qmgr](#):

```
> qmgr -c 'set server mail_body_fmt=Job: %i \n Name: %j \n On host: %h \n \n %m \n \n %d'
> qmgr -c 'set server mail_subject_fmt=Job %i - %r'
```

By default, users receive e-mails on job aborts. Each user can select which kind of e-mails to receive by using the `qsub -m` option when submitting the job. If you want to dictate when each user should receive e-mails, use a submit filter (for details, see [Appendix J: Job submission filter \("qsub wrapper"\) on page 2827](#)).

## Using MUNGE authentication

MUNGE is an authentication service that creates and validates user credentials. It was developed by Lawrence Livermore National Laboratory (LLNL) to be highly scalable so it can be used in large environments such as HPC clusters. To learn more about MUNGE and how to install it, see <http://code.google.com/p/munge/>.

Configuring TORQUE to use MUNGE is a compile time operation. When you are building TORQUE, use `-enable-munge-auth` as a command line option with `./configure`.

```
> ./configure -enable-munge-auth
```

You can use only one authorization method at a time. If `-enable-munge-auth` is configured, the privileged port `ruserok` method is disabled.

TORQUE does not link any part of the MUNGE library into its executables. It calls the MUNGE and UNMUNGE utilities which are part of the MUNGE daemon. The MUNGE daemon must be running on the server and all submission hosts. The TORQUE client utilities call MUNGE and then deliver the encrypted credential to `pbs_server` where the credential is then unmunged and the server verifies the user and host against the authorized users configured in `serverdb`.

Authorized users are added to `serverdb` using `qmgr` and the **authorized\_users** parameter. The syntax for **authorized\_users** is `authorized_users=<user>@<host>`. To add an authorized user to the server you can use the following `qmgr` command:

```
> qmgr -c 'set server authorized_users=user1@hosta
> qmgr -c 'set server authorized_users+=user2@hosta'
```

The previous example adds `user1` and `user2` from `hosta` to the list of authorized users on the server. Users can be removed from the list of authorized users by using the `-=` syntax as follows:

```
> qmgr -c 'set server authorized_users-=user1@hosta'
```

Users must be added with the `<user>@<host>` syntax. The user and the host portion can use the `*` wildcard to allow multiple names to be accepted with a single entry. A range of user or host names can be specified using a `[a-b]` syntax where *a* is the beginning of the range and *b* is the end.

```
> qmgr -c 'set server authorized_users=user[1-10]@hosta'
```

This allows `user1` through `user10` on `hosta` to run client commands on the server.

## Setting up the MOM hierarchy

The MOM hierarchy allows you to override the compute nodes' default behavior of reporting status updates directly to the `pbs_server`. Instead, you configure compute nodes so that each node sends its status update information to another compute node. The compute nodes pass the information up a tree or hierarchy until eventually the information reaches a node that will pass the information directly to `pbs_server`. This can significantly reduce traffic and time required to keep the cluster status up to date.

The name of the file that contains the configuration information is named `mom_hierarchy`. By default, it is located in the `/var/spool/torque/server_priv` directory. The file uses syntax similar to XML:

```
<path>
  <level> comma-separated node list </level>
  <level> comma-separated node list </level>
  ...
</path>
...
```

The `<path></path>` tag pair identifies a group of compute nodes. The `<level></level>` tag pair contains a comma-separated list of compute node names. Multiple paths can be defined with multiple levels within each path.

Within a `<path>` tag pair the levels define the hierarchy. All nodes in the top level communicate directly with the server. All nodes in lower levels communicate to the first available node in the level directly above it. If the first node in the upper level goes down, the nodes in the subordinate level will then communicate to the next node in the upper level. If no nodes are available in an upper level then the node will communicate directly to the server.

If an upper level node has fallen out and then becomes available, the lower level nodes will eventually find that the node is available and start sending their updates to that node.

**i** If you want to specify MOMs on a different port than the default, you must list the node in the form: `hostname:mom_manager_port`.

For example:

```
<path>
  <level>hostname:mom_manager_port</level>
  ...
</path>
...
```

## Putting the MOM hierarchy on the MOMs

You can put the MOM hierarchy file directly on the MOMs. This way, the `pbs_server` doesn't have to send the hierarchy to all the MOMs during each `pbs_server` startup. The hierarchy file still has to exist on the `pbs_server` and if the file versions conflict, the `pbs_server` version overwrites the local MOM file. Due to this, it is recommended that the hierarchy file either be symlinked to the MOMs or put on a global NFS share.

Once the hierarchy file exists on the MOMs, start `pbs_server` with the `-n` option which tells `pbs_server` to not send the hierarchy file on startup. Instead, `pbs_server` waits until a MOM requests it.

### Related topics

- [Advanced configuration on page 2558](#)

## Manual setup of initial server configuration

On a new installation of TORQUE, the server database must be initialized using the command `pbs_server -t create`. This command creates a file in `$TORQUEHOME/server_priv` named `serverdb` which contains the server configuration information.

The following output from `qmgr` shows the base configuration created by the command `pbs_server -t create`:

```
qmgr -c 'p s'
#
Set server attributes.
#
set server acl_hosts = kmn
set server log_events = 511
set server mail_from = adm
set server node_check_rate = 150
set server tcp_timeout = 6
```

This is a bare minimum configuration and it is not very useful. By using `qmgr`, the server configuration can be modified to set up TORQUE to do useful work. The following `qmgr` commands will create a queue and enable the server to accept and run jobs. These commands must be executed by root.

```
pbs_server -t create
qmgr -c "set server scheduling=true"
qmgr -c "create queue batch queue_type=execution"
qmgr -c "set queue batch started=true"
qmgr -c "set queue batch enabled=true"
qmgr -c "set queue batch resources_default.nodes=1"
qmgr -c "set queue batch resources_default.walltime=3600"
qmgr -c "set server default_queue=batch"
```



When TORQUE reports a new queue to Moab a class of the same name is automatically applied to all nodes.

In this example, the configuration database is initialized and the scheduling interface is activated using (`scheduling=true`). This option allows the scheduler to receive job and node events which allow it to be more responsive (See [scheduling on page 2782](#) for more information). The next command creates a queue and specifies the queue type. Within PBS, the queue must be declared an `execution` queue in order for it to run jobs. Additional configuration (i.e., setting the queue to `started` and `enabled`) allows the queue to *accept* job submissions, and *launch* queued jobs.

The next two lines are optional, setting default `node` and `walltime` attributes for a submitted job. These defaults will be picked up by a job if values are not explicitly set by the submitting user. The final line, `default_queue=batch`, is also a convenience line and indicates that a job should be placed in the batch queue unless explicitly assigned to another queue.

Additional information on configuration can be found in the admin manual and in the [qmgr](#) main page.

## Related topics

- [TORQUE installation overview on page 2544](#)

## Server node file configuration

This section contains information about configuring server node files. It explains how to specify node virtual processor counts and GPU counts, as well as how to specify node features or properties. For details, see these topics:

- [Basic node specification on page 2572](#)
- [Specifying virtual processor count for a node on page 2572](#)
- [Specifying GPU count for a node on page 2573](#)
- [Specifying node features \(node properties\) on page 2573](#)

## Related topics

- [TORQUE installation overview on page 2544](#)
- [Appendix B: Server parameters on page 2766](#)
- [Moab node feature overview](#)

## Basic node specification

For the `pbs_server` to communicate with each of the MOMs, it needs to know which machines to contact. Each node that is to be a part of the batch system must be specified on a line in the `server nodes` file. This file is located at `TORQUE_HOME/server_priv/nodes`. In most cases, it is sufficient to specify just the node name on a line as in the following example:

`server_priv/nodes:`

```
node001
node002
node003
node004
```

## Related topics

- [Server node file configuration on page 2572](#)

## Specifying virtual processor count for a node

By default each node has one virtual processor. Increase the number using the **np** attribute in the nodes file. The value of `np` can be equal to the number of physical cores on the node or it can be set to a value which represents available "execution slots" for the node. The value used is determined by the administrator based on hardware, system, and site criteria.

The following example shows how to set the `np` value in the nodes file. In this example, we are assuming that `node001` and `node002` have four physical cores. The administrator wants the value of `np` for `node001`

to reflect that it has four cores. However, node002 will be set up to handle multiple virtual processors without regard to the number of physical cores on the system.

server\_priv/nodes:

```
node001 np=4
node002 np=12
...
```

#### Related topics

- [Server node file configuration on page 2572](#)

## Specifying GPU count for a node

Administrators can manually set the number of GPUs on a node or if they are using NVIDIA GPUs and drivers, they can have them detected automatically. For more information about how to set up TORQUE with GPUS, see [Accelerators on page 815](#).

To manually set the number of GPUs on a node, use the `gpus` attribute in the nodes file. The value of GPUs is determined by the administrator based on hardware, system, and site criteria.

The following example shows how to set the GPU value in the nodes file. In the example, we assume node001 and node002 each have two physical GPUs. The administrator wants the value of node001 to reflect the physical GPUs available on that system and adds `gpus=2` to the nodes file entry for node001. However, node002 will be set up to handle multiple virtual GPUs without regard to the number of physical GPUs on the system.

server\_priv/nodes:

```
node001 gpus=1
node002 gpus=4
...
```

#### Related topics

- [Server node file configuration on page 2572](#)

## Specifying node features (node properties)

Node features can be specified by placing one or more white space-delimited strings on the line for the associated host as in the following example:

server\_priv/nodes:

```
node001 np=2 fast ia64
node002 np=4 bigmem fast ia64 smp
...
```

These features can be used by users to request specific nodes when submitting jobs. For example:

```
qsub -l nodes=1:bigmem+1:fast job.sh
```

This job submission will look for a node with the bigmem feature (node002) and a node with the fast feature (either node001 or node002).

## Related topics

- [Server node file configuration on page 2572](#)

## Testing server configuration

If you have initialized TORQUE using the `torque.setup` script or started TORQUE using `pbs_server -t create` and `pbs_server` is still running, terminate the server by calling `qterm`. Next, start `pbs_server` again without the `-t create` arguments. Follow the script below to verify your server configuration. The output for the examples below is based on the nodes file example in [Specifying node features](#) and [Server configuration](#).

```

# verify all queues are properly configured
> qstat -q

server:kmn

Queue   Memory   CPU Time   Walltime   Node   Run   Que   Lm   State
-----
batch   --       --         --         --     0     0    --   ER
                                     ---
                                     0     0

# view additional server configuration
> qmgr -c 'p s'
#
# Create queues and set their attributes
#
# Create and define queue batch
#
create queue batch
set queue batch queue_type = Execution
set queue batch resources_default.nodes = 1
set queue batch resources_default.walltime = 01:00:00
set queue batch enabled = True
set queue batch started = True
#
# Set server attributes.
#
set server scheduling = True
set server acl_hosts = kmn
set server managers = user1@kmn
set server operators = user1@kmn
set server default_queue = batch
set server log_events = 511
set server mail_from = adm
set server node_check_rate = 150
set server tcp_timeout = 300
set server job_stat_rate = 45
set server poll_jobs = True
set server mom_job_sync = True
set server keep_completed = 300
set server next_job_number = 0

# verify all nodes are correctly reporting
> pbsnodes -a
node001
  state=free
  np=2
  properties=bigmem,fast,ia64,smp
  ntype=cluster
  status=rectime=1328810402,varattr=,jobs=,state=free,netload=6814326158,gres=,loadave
=0.21,ncpus=6,physmem=8193724kb,
availmem=13922548kb,totmem=16581304kb,idletime=3,nusers=3,nsessions=18,sessions=1876
1120 1912 1926 1937 1951 2019 2057 28399 2126 2140 2323 5419 17948 19356 27726 22254
29569,uname=Linux kmn 2.6.38-11-generic #48-Ubuntu SMP Fri Jul 29 19:02:55 UTC 2011
x86_64,opsys=linux
  mom_service_port = 15002
  mom_manager_port = 15003
  gpus = 0
# submit a basic job - DO NOT RUN AS ROOT
> su - testuser
> echo "sleep 30" | qsub

# verify jobs display
> qstat

Job id   Name   User   Time Use   S   Queue
-----

```

```
0.kmn STDIN knielson 0 Q batch
```

At this point, the job should be in the **Q** state and will not run because a scheduler is not running yet. TORQUE can use its native scheduler by running `pbs_sched` or an advanced scheduler (such as Moab Workload Manager). See [Integrating schedulers for TORQUE on page 2653](#) for details on setting up an advanced scheduler.

#### Related topics

- [TORQUE installation overview on page 2544](#)

## TORQUE on NUMA systems

Starting in TORQUE version 3.0, TORQUE can be configured to take full advantage of Non-Uniform Memory Architecture (NUMA) systems. The following instructions are a result of development on SGI Altix and UV hardware.

For details, see these topics:

- [TORQUE NUMA configuration on page 2576](#)
- [Building TORQUE with NUMA support on page 2576](#)

### TORQUE NUMA configuration

There are three steps to configure TORQUE to take advantage of NUMA architectures:

1. Configure TORQUE with `--enable-numa-support`.
2. Create the `mom_priv/mom.layout` file.
3. Configure `server_priv/nodes`.

#### Related topics

- [TORQUE on NUMA systems on page 2576](#)

### Building TORQUE with NUMA support

To turn on NUMA support for TORQUE the `--enable-numa-support` option must be used during the configure portion of the installation. In addition to any other configuration options, add the `--enable-numa-support` option as indicated in the following example:

```
$ ./configure --enable-numa-support
```

 Don't use MOM hierarchy with NUMA.

When TORQUE is enabled to run with NUMA support, there is only a single instance of `pbs_mom` (MOM) that is run on the system. However, TORQUE will report that there are multiple nodes running in the cluster. While `pbs_mom` and `pbs_server` both know there is only one instance of `pbs_mom`, they manage the cluster as if there were multiple separate MOM nodes.

The `mom.layout` file is a virtual mapping between the system hardware configuration and how the administrator wants TORQUE to view the system. Each line in `mom.layout` equates to a node in the cluster and is referred to as a NUMA node.

## Automatically Creating `mom.layout` (Recommended)

A perl script named `mom_gencfg` is provided in the `contrib/` directory that generates the `mom.layout` file for you. The script can be customized by setting a few variables in it. To automatically create the `mom.layout` file, follow these instructions (these instructions are also included in the script):

1. Verify `hwloc` version 1.1 or higher is installed - see `contrib/hwloc_install.sh`.
2. Install `Sys::Hwloc` from CPAN.
3. Verify `$PBS_HOME` is set to the proper value.
4. Update the variables in the 'Config Definitions' section of the script. Especially update `firstNodeId` and `nodesPerBoard` if desired. The `firstNodeId` variable should be set above 0 if you have a root `cpuset` that you wish to exclude and the `nodesPerBoard` variable is the number of NUMA nodes per board. Each node is defined in `/sys/devices/system/node`, in a subdirectory `node<node index>`
5. Back up your current file in case a variable is set incorrectly or neglected.
6. Run the script.

7. 

```
$ ./mom_gencfg
```

## Manually Creating `mom.layout`

To properly set up the `mom.layout` file, it is important to know how the hardware is configured. Use the `topology` command line utility and inspect the contents of `/sys/devices/system/node`. The `hwloc` library can also be used to create a custom discovery tool.

Typing `topology` on the command line of a NUMA system produces something similar to the following:

```

Partition number: 0
6 Blades
72 CPUs
378.43 Gb Memory Total

```

Blade	ID	asic	NASID	Memory
0	r001i01b00	UVHub 1.0	0	67089152 kB
1	r001i01b01	UVHub 1.0	2	67092480 kB
2	r001i01b02	UVHub 1.0	4	67092480 kB
3	r001i01b03	UVHub 1.0	6	67092480 kB
4	r001i01b04	UVHub 1.0	8	67092480 kB
5	r001i01b05	UVHub 1.0	10	67092480 kB

CPU	Blade	PhysID	CoreID	APIC-ID	Family	Model	Speed	L1 (KiB)	L2 (KiB)	L3 (KiB)
0	r001i01b00	00	00	0	6	46	2666	32d/32i	256	18432
1	r001i01b00	00	02	4	6	46	2666	32d/32i	256	18432
2	r001i01b00	00	03	6	6	46	2666	32d/32i	256	18432
3	r001i01b00	00	08	16	6	46	2666	32d/32i	256	18432
4	r001i01b00	00	09	18	6	46	2666	32d/32i	256	18432
5	r001i01b00	00	11	22	6	46	2666	32d/32i	256	18432
6	r001i01b00	01	00	32	6	46	2666	32d/32i	256	18432
7	r001i01b00	01	02	36	6	46	2666	32d/32i	256	18432
8	r001i01b00	01	03	38	6	46	2666	32d/32i	256	18432
9	r001i01b00	01	08	48	6	46	2666	32d/32i	256	18432
10	r001i01b00	01	09	50	6	46	2666	32d/32i	256	18432
11	r001i01b00	01	11	54	6	46	2666	32d/32i	256	18432
12	r001i01b01	02	00	64	6	46	2666	32d/32i	256	18432
13	r001i01b01	02	02	68	6	46	2666	32d/32i	256	18432
14	r001i01b01	02	03	70	6	46	2666	32d/32i	256	18432

From this partial output, note that this system has 72 CPUs on 6 blades. Each blade has 12 CPUs grouped into clusters of 6 CPUs. If the entire content of this command were printed you would see each Blade ID and the CPU ID assigned to each blade.

The topology command shows how the CPUs are distributed, but you likely also need to know where memory is located relative to CPUs, so go to `/sys/devices/system/node`. If you list the node directory you will see something similar to the following:

```

# ls -al
total 0
drwxr-xr-x 14 root root 0 Dec 3 12:14 .
drwxr-xr-x 14 root root 0 Dec 3 12:13 ..
-r--r--r-- 1 root root 4096 Dec 3 14:58 has_cpu
-r--r--r-- 1 root root 4096 Dec 3 14:58 has_normal_memory
drwxr-xr-x 2 root root 0 Dec 3 12:14 node0
drwxr-xr-x 2 root root 0 Dec 3 12:14 node1
drwxr-xr-x 2 root root 0 Dec 3 12:14 node10
drwxr-xr-x 2 root root 0 Dec 3 12:14 node11
drwxr-xr-x 2 root root 0 Dec 3 12:14 node2
drwxr-xr-x 2 root root 0 Dec 3 12:14 node3
drwxr-xr-x 2 root root 0 Dec 3 12:14 node4
drwxr-xr-x 2 root root 0 Dec 3 12:14 node5
drwxr-xr-x 2 root root 0 Dec 3 12:14 node6
drwxr-xr-x 2 root root 0 Dec 3 12:14 node7
drwxr-xr-x 2 root root 0 Dec 3 12:14 node8
drwxr-xr-x 2 root root 0 Dec 3 12:14 node9
-r--r--r-- 1 root root 4096 Dec 3 14:58 online
-r--r--r-- 1 root root 4096 Dec 3 14:58 possible

```

The directory entries `node0`, `node1`, ..., `node11` represent groups of memory and CPUs local to each other. These groups are a node board, a grouping of resources that are close together. In most cases, a node board is made up of memory and processor cores. Each bank of memory is called a memory node by the

operating system, and there are certain CPUs that can access that memory very rapidly. Note under the directory for node board node0 that there is an entry called `cpulist`. This contains the CPU IDs of all CPUs local to the memory in node board 0.

Now create the `mom.layout` file. The content of `cpulist` 0-5 are local to the memory of node board 0, and the memory and `cpus` for that node are specified in the layout file by saying `nodes=0`. The `cpulist` for node board 1 shows 6-11 and memory node index 1. To specify this, simply write `nodes=1`. Repeat this for all twelve node boards and create the following `mom.layout` file for the 72 CPU system.

```
nodes=0
nodes=1
nodes=2
nodes=3
nodes=4
nodes=5
nodes=6
nodes=7
nodes=8
nodes=9
nodes=10
nodes=11
```

Each line in the `mom.layout` file is reported as a node to `pbs_server` by the `pbs_mom` daemon.

The `mom.layout` file does not need to match the hardware layout exactly. It is possible to combine node boards and create larger NUMA nodes. The following example shows how to do this:

```
nodes=0-1
```

The memory nodes can be combined the same as CPUs. The memory nodes combined must be contiguous. You cannot combine mem 0 and 2.

## Configuring `server_priv/nodes`

The `pbs_server` requires awareness of how the MOM is reporting nodes since there is only one MOM daemon and multiple MOM nodes. So, configure the `server_priv/nodes` file with the **`num_node_boards`** and **`numa_board_str`** attributes. The attribute `num_node_boards` tells `pbs_server` how many numa nodes are reported by the MOM. Following is an example of how to configure the `nodes` file with `num_node_boards`:

```
numa-10 np=72 num_node_boards=12
```

This line in the `nodes` file tells `pbs_server` there is a host named `numa-10` and that it has 72 processors and 12 nodes. The `pbs_server` divides the value of `np` (72) by the value for `num_node_boards` (12) and determines there are 6 CPUs per NUMA node.

In this example, the NUMA system is uniform in its configuration of CPUs per node board, but a system does not need to be configured with the same number of CPUs per node board. For systems with non-uniform CPU distributions, use the attribute **`numa_board_str`** to let `pbs_server` know where CPUs are located in the cluster.

The following is an example of how to configure the `server_priv/nodes` file for non-uniformly distributed CPUs:

```
Numa-11 numa_board_str=6,8,12
```

In this configuration, `pbs_server` knows it has three MOM nodes and the nodes have 6, 8, and 12 CPUs respectively. Note that the attribute `np` is not used. The `np` attribute is ignored because the number of CPUs per node is expressly given.

## Enforcement of memory resource limits

TORQUE can better enforce memory limits with the use of the utility **memacctd**. The `memacctd` utility is provided by SGI on SuSe Linux Enterprise Edition (SLES). It is a daemon that caches memory footprints when it is queried. When configured to use the memory monitor, TORQUE queries `memacctd`. It is up to the user to make sure `memacctd` is installed. See the [SGI memacctd man page](#) for more information.

### To configure TORQUE to use memacctd for memory enforcement

1. Start **memacctd** as instructed by SGI.
2. Reconfigure TORQUE with `--enable-memacct`. This will link in the necessary library when TORQUE is recompiled.
3. Recompile and reinstall TORQUE.
4. Restart all MOM nodes.
5. (Optional) Alter the [gsub](#) filter to include a default memory limit for all jobs that are not submitted with memory limit.

#### Related topics

- [TORQUE NUMA configuration on page 2576](#)
- [TORQUE on NUMA systems on page 2576](#)

## TORQUE Multi-MOM

Starting in TORQUE version 3.0 users can run multiple MOMs on a single node. The initial reason to develop a multiple MOM capability was for testing purposes. A small cluster can be made to look larger since each MOM instance is treated as a separate node.

When running multiple MOMs on a node each MOM must have its own service and manager ports assigned. The default ports used by the MOM are 15002 and 15003. With the multi-mom alternate ports can be used without the need to change the default ports for `pbs_server` even when running a single instance of the MOM.

For details, see these topics:

- [Multi-MOM configuration on page 2580](#)
- [Stopping pbs\\_mom in Multi-MOM mode on page 2582](#)

## Multi-MOM configuration

There are three steps to setting up multi-MOM capability:

1. [Configure server\\_priv/nodes on page 2581](#)
2. [/etc/hosts file on page 2581](#)
3. [Starting pbs\\_mom with multi-MOM options on page 2581](#)

## Configure server\_priv/nodes

The attributes **mom\_service\_port** and **mom\_manager\_port** were added to the nodes file syntax to accommodate multiple MOMs on a single node. By default `pbs_mom` opens ports 15002 and 15003 for the service and management ports respectively. For multiple MOMs to run on the same IP address they need to have their own port values so they can be distinguished from each other. `pbs_server` learns about the port addresses of the different MOMs from entries in the `server_priv/nodes` file. The following is an example of a nodes file configured for multiple MOMs:

```
hosta    np=2
hosta-1  np=2 mom_service_port=30001 mom_manager_port=30002
hosta-2  np=2 mom_service_port=31001 mom_manager_port=31002
hosta-3  np=2 mom_service_port=32001 mom_manager_port=32002
```

Note that all entries have a unique host name and that all port values are also unique. The entry `hosta` does not have a `mom_service_port` or `mom_manager_port` given. If unspecified, then the MOM defaults to ports 15002 and 15003.

## /etc/hosts file

Host names in the `server_priv/nodes` file must be resolvable. Creating an alias for each host enables the server to find the IP address for each MOM; the server uses the port values from the `server_priv/nodes` file to contact the correct MOM. An example `/etc/hosts` entry for the previous `server_priv/nodes` example might look like the following:

```
192.65.73.10 hosta hosta-1 hosta-2 hosta-3
```

Even though the host name and all the aliases resolve to the same IP address, each MOM instance can still be distinguished from the others because of the unique port value assigned in the `server_priv/nodes` file.

## Starting pbs\_mom with multi-MOM options

To start multiple instances of `pbs_mom` on the same node, use the following syntax (see [pbs\\_mom on page 2693](#) for details):

```
pbs_mom -m -M <port value of MOM_service_port> -R <port value of MOM_manager_port> -A
<name of MOM alias>
```

Continuing based on the earlier example, if you want to create four MOMs on `hosta`, type the following at the command line:

```
# pbs_mom -m -M 30001 -R 30002 -A hosta-1
# pbs_mom -m -M 31001 -R 31002 -A hosta-2
# pbs_mom -m -M 32001 -R 32002 -A hosta-3
# pbs_mom
```

Notice that the last call to `pbs_mom` uses no arguments. By default `pbs_mom` opens on ports 15002 and 15003. No arguments are necessary because there are no conflicts.

#### Related topics

- [TORQUE Multi-MOM on page 2580](#)
- [Stopping pbs\\_mom in Multi-MOM mode on page 2582](#)

## Stopping pbs\_mom in Multi-MOM mode

Terminate `pbs_mom` by using the `momctl -s` command (for details, see [momctl](#)). For any MOM using the default manager port 15003, the `momctl -s` command stops the MOM. However, to terminate MOMs with a manager port value not equal to 15003, you must use the following syntax:

```
momctl -s -p <port value of MOM_manager_port>
```

The `-p` option sends the terminating signal to the MOM manager port and the MOM is terminated.

#### Related topics

- [TORQUE Multi-MOM on page 2580](#)
- [Multi-MOM configuration on page 2580](#)

# Submitting and managing jobs

This section contains information about how you can submit and manage jobs with TORQUE. For details, see the following topics:

- [Job submission on page 2583](#)
- [Monitoring jobs on page 2599](#)
- [Canceling jobs on page 2599](#)
- [Job preemption on page 2600](#)
- [Keeping completed jobs on page 2600](#)
- [Job checkpoint and restart on page 2601](#)
- [Job exit status on page 2611](#)
- [Service jobs on page 2615](#)

## Job submission

Job submission is accomplished using the **qsub** command, which takes a number of command line arguments and integrates such into the specified PBS command file. The PBS command file may be specified as a filename on the **qsub** command line or may be entered via STDIN.

- The PBS command file does not need to be executable.
- The PBS command file may be *piped* into **qsub** (i.e., `cat pbs.cmd | qsub`).
- In the case of parallel jobs, the PBS command file is staged to, and executed on, the first allocated compute node only. (Use **pbsdsh** to run actions on multiple nodes.)
- The command script is executed from the user's home directory in all cases. (The script may determine the submission directory by using the `$PBS_O_WORKDIR` environment variable)
- The command script will be executed using the default set of user environment variables unless the **-V** or **-v** flags are specified to include aspects of the job submission environment.
- PBS directives should be declared first in the job script.

```
#PBS -S /bin/bash
#PBS -m abe
#PBS -M <yourEmail@company.com>
echo sleep 300
```

*This is an example of properly declared PBS directives.*

```
#PBS -S /bin/bash
SOMEVARIABLE=42
#PBS -m abe
#PBS -M <yourEmail@company.com>
echo sleep 300
```

*This is an example of improperly declared PBS directives. PBS directives below "SOMEVARIABLE=42" are ignored.*

**i** By default, job submission is allowed only on the TORQUE server host (host on which **pbs\_server** is running). Enablement of job submission from other hosts is documented in [Server configuration on page 2566](#).

**i** Versions of TORQUE earlier than 2.4.5 attempted to apply queue and server defaults to a job that didn't have defaults specified. If a setting still did not have a value after that, TORQUE applied the queue and server maximum values to a job (meaning, the maximum values for an applicable setting were applied to jobs that had no specified or default value).

In TORQUE 2.4.5 and later, the queue and server maximum values are no longer used as a value for missing settings.

This section contains these topics:

- [Multiple job submission on page 2584](#)
- [Requesting resources on page 2586](#)

- [Requesting generic resources](#) on page 2593
- [Requesting floating resources](#) on page 2593
- [Requesting other resources](#) on page 2594
- [Exported batch environment variables](#) on page 2594
- [Enabling trusted submit hosts](#) on page 2596
- [Example submit scripts](#) on page 2596

### Related topics

- Maui Documentation
- <http://www.lunarc.lu.se>
- [http://www.clusters.umaine.edu/wiki/index.php/Example\\_Submission\\_Scripts](http://www.clusters.umaine.edu/wiki/index.php/Example_Submission_Scripts)
- [Appendix J: Job submission filter \("qsub wrapper"\)](#) on page 2827 – Allow local checking and modification of submitted job

## Multiple job submission

Sometimes users will want to submit large numbers of jobs based on the same job script. Rather than using a script to repeatedly call `qsub`, a feature known as job arrays now exists to allow the creation of multiple jobs with one `qsub` command. Additionally, this feature includes a new job naming convention that allows users to reference the entire set of jobs as a unit, or to reference one particular job from the set.

Job arrays are submitted through the `-t` option to `qsub`, or by using `#PBS -t` in your batch script. This option takes a comma-separated list consisting of either a single job ID number, or a pair of numbers separated by a dash. Each of these jobs created will use the same script and will be running in a nearly identical environment.

```
> qsub -t 0-4 job_script
1098[0].hostname

> qstat -t
1098[0].hostname ...
1098[1].hostname ...
1098[2].hostname ...
1098[3].hostname ...
1098[4].hostname ...
```

**i** Versions of TORQUE earlier than 2.3 had different semantics for the `-t` argument. In these versions, `-t` took a single integer number—a count of the number of jobs to be created.

Each `1098[x]` job has an environment variable called `PBS_ARRAYID`, which is set to the value of the array index of the job, so `1098[0].hostname` would have `PBS_ARRAYID` set to 0. This allows you to create job arrays where each job in the array performs slightly different actions based on the value of this variable, such as performing the same tasks on different input files. One other difference in the environment between jobs in the same array is the value of the `PBS_JOBNAME` variable.

```
# These two examples are equivalent in TORQUE 2.2
> qsub -t 0-99
> qsub -t 100

# You can also pass comma delimited lists of ids and ranges:
> qsub -t 0,10,20,30,40
> qsub -t 0-50,60,70,80
```

Running `qstat` displays a job summary, which provides an overview of the array's state. To see each job in the array, run `qstat -t`.

The `qalter`, `qdel`, `qhold`, and `qrls` commands can operate on arrays—either the entire array or a range of that array. Additionally, any job in the array may be accessed normally by using that job's ID, just as you would with any other job. For example, running the following command would run only the specified job:

```
qrun 1098[0].hostname
```

## Slot Limit

The slot limit is a way for administrators to limit the number of jobs from a job array that can be eligible for scheduling at the same time. When a slot limit is used, TORQUE puts a hold on all jobs in the array that exceed the slot limit. When an eligible job in the array completes, TORQUE removes the hold flag from the next job in the array. Slot limits can be declared globally with the [max\\_slot\\_limit](#) parameter, or on a per-job basis with [qsub -t](#).

Related topics

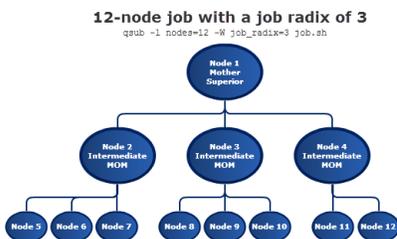
- [Job submission on page 2583](#)

## Managing multi-node jobs

By default, when a multi-node job runs, the Mother Superior manages the job across all the sister nodes by communicating with each of them and updating `pbs_server`. Each of the sister nodes sends its updates and stdout and stderr directly to the Mother Superior. When you run an extremely large job using hundreds or thousands of nodes, you may want to reduce the amount of network traffic sent from the sisters to the Mother Superior by specifying a job radix. Job radix sets a maximum number of nodes with which the Mother Superior and resulting intermediate MOMs communicate and is specified using the [-W on page 2755](#) option for `qsub`.

For example, if you submit a smaller, 12-node job and specify `job_radix=3`, Mother Superior and each resulting intermediate MOM is only allowed to receive communication from 3 subordinate nodes.

Image 4-1: Job radix example



The Mother Superior picks three sister nodes with which to communicate the job information. Each of those nodes (intermediate MOMs) receives a list of all sister nodes that will be subordinate to it. They each contact up to three nodes and pass the job information on to those nodes. This pattern continues until the bottom level is reached. All communication is now passed across this new hierarchy. The stdout and stderr data is aggregated and sent up the tree until it reaches the Mother Superior, where it is saved and copied to the `.o` and `.e` files.

**i** Job radix is meant for extremely large jobs only. It is a tunable parameter and should be adjusted according to local conditions in order to produce the best results.

## Requesting resources

Various resources can be requested at the time of job submission. A job can request a particular node, a particular node attribute, or even a number of nodes with particular attributes. Either native TORQUE resources or external scheduler resource extensions may be specified. The native TORQUE resources are listed in the following table:

Resource	Format	Description
<b>arch</b>	string	Specifies the administrator defined system architecture required. This defaults to whatever the <b>PBS_MACH</b> string is set to in "local.mk".
<b>cpus</b>	seconds, or [[HH:]MM:]SS	Maximum amount of CPU time used by all processes in the job.

Resource	Format	Description
<b>cpuclock</b>	string	<p>Specify the CPU clock frequency for each node requested for this job. A <b>cpuclock</b> request applies to every processor on every node in the request. Specifying varying CPU frequencies for different nodes or different processors on nodes in a single job request is not supported.</p> <p>Not all processors support all possible frequencies or ACPI states. If the requested frequency is not supported by the CPU, the nearest frequency is used.</p> <p>ALPS 1.4 or later is required when using <b>cpuclock</b> on Cray.</p> <p>The clock frequency can be specified via:</p> <ul style="list-style-type: none"> <li>a number that indicates the clock frequency (with or without the SI unit suffix).</li> </ul> <pre>qsub -l cpuclock=1800,nodes=2 script.sh qsub -l cpuclock=1800mhz,nodes=2 script.sh</pre> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to 1800 MHz.</i></p> <ul style="list-style-type: none"> <li>a Linux power governor policy name. The governor names are: <ul style="list-style-type: none"> <li><b>performance</b>: This governor instructs Linux to operate each logical processor at its maximum clock frequency. This setting consumes the most power and workload executes at the fastest possible speed.</li> <li><b>powersave</b>: This governor instructs Linux to operate each logical processor at its minimum clock frequency. This setting executes workload at the slowest possible speed. This setting does not necessarily consume the least amount of power since applications execute slower, and may actually consume more energy because of the additional time needed to complete the workload's execution.</li> <li><b>ondemand</b>: This governor dynamically switches the logical processor's clock frequency to the maximum value when system load is high and to the minimum value when the system load is low. This setting causes workload to execute at the fastest possible speed or the slowest possible speed, depending on OS load. The system switches between consuming the most power and the least power.</li> </ul> </li> </ul>

Resource	Format	Description
		<div data-bbox="764 317 1409 709" style="border: 1px solid #ccc; padding: 10px; margin-bottom: 10px;"> <p><b>i</b> The power saving benefits of <i>ondemand</i> might be non-existent due to frequency switching latency if the system load causes clock frequency changes too often. This has been true for older processors since changing the clock frequency required putting the processor into the C3 "sleep" state, changing its clock frequency, and then waking it up, all of which required a significant amount of time. Newer processors, such as the Intel Xeon E5-2600 Sandy Bridge processors, can change clock frequency dynamically and much faster.</p> </div> <ul style="list-style-type: none"> <li data-bbox="732 720 1409 947"> <p>o <i>conservative</i>: This governor operates like the <i>ondemand</i> governor but is more conservative in switching between frequencies. It switches more gradually and uses all possible clock frequencies. This governor can switch to an intermediate clock frequency if it seems appropriate to the system load and usage, which the <i>ondemand</i> governor does not do.</p> </li> </ul> <div data-bbox="683 961 1409 1010" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <pre>qsub -l cpuclock=performance,nodes=2 script.sh</pre> </div> <div data-bbox="732 1024 1360 1108" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to the performance power governor policy.</i></p> </div> <ul style="list-style-type: none"> <li data-bbox="651 1136 1409 1346"> <p>• an ACPI performance state (or P-state) with or without the P prefix. P-states are a special range of values (0-15) that map to specific frequencies. Not all processors support all 16 states, however, they all start at P0. P0 sets the CPU clock frequency to the highest performance state which runs at the maximum frequency. P15 sets the CPU clock frequency to the lowest performance state which runs at the lowest frequency.</p> </li> </ul> <div data-bbox="683 1371 1409 1440" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <pre>qsub -l cpuclock=3,nodes=2 script.sh qsub -l cpuclock=p3,nodes=2 script.sh</pre> </div> <div data-bbox="732 1455 1360 1539" style="border: 1px dashed #ccc; padding: 5px; margin-bottom: 10px;"> <p><i>This job requests 2 nodes and specifies their CPU frequencies should be set to a performance state of 3.</i></p> </div> <p>When reviewing job or node properties when <b>cpuclock</b> was used, be mindful of unit conversion. The OS reports frequency in Hz, not MHz or GHz.</p>

Resource	Format	Description
<b>epilogue</b>	string	Specifies a user owned epilogue script which will be run before the system epilogue and epilogue.user scripts at the completion of a job. The syntax is <code>epilogue=&lt;file&gt;</code> . The file can be designated with an absolute or relative path. For more information, see <a href="#">Appendix G: Prologue and epilogue scripts on page 2817</a> .
<b>feature</b>	string	Specifies a property or feature for the job. Feature corresponds to TORQUE node properties and Moab features. <pre>qsub script.sh -l procs=10,feature=bigmem</pre>
<b>file</b>	<a href="#">size</a>	The amount of total disk requested for the job. (Ignored on Unicos.)
<b>host</b>	string	Name of the host on which the job should be run. This resource is provided for use by the site's scheduling policy. The allowable values and effect on job placement is site dependent.
<b>mem</b>	<a href="#">size</a>	Maximum amount of physical memory used by the job. Ignored on Darwin, Digital Unix, Free BSD, HPUX 11, IRIX, NetBSD, and SunOS. Not implemented on AIX and HPUX 10. The <code>mem</code> resource will only work for single-node jobs. If your job requires multiple nodes, use <code>pmem</code> instead.
<b>ncpus</b>	integer	The number of processors in one task where a task cannot span nodes.  You cannot request both <code>ncpus</code> and <code>nodes</code> in the same job.
<b>nice</b>	integer	Number between -20 (highest priority) and 19 (lowest priority). Adjust the process execution priority.

Resource	Format	Description
<b>nodes</b>	{<node_count>   <hostname>} [:ppn=<ppn>] [:gpus=<gpu>] [:<property>] [:<property>...] [+ ...]	<p>Number and/or type of nodes to be reserved for exclusive use by the job. The value is one or more node_specs joined with the + (plus) character: node_spec [+node_spec...]. Each node_spec is a number of nodes required of the type declared in the node_spec and a name of one or more properties desired for the nodes. The number, the name, and each property in the node_spec are separated by a : (colon). If no number is specified, one (1) is assumed. The name of a node is its hostname. The properties of nodes are:</p> <ul style="list-style-type: none"> <li>• <b>ppn=#</b> - Specify the number of virtual processors per node requested for this job. The number of virtual processors available on a node by default is 1, but it can be configured in the \$TORQUE_HOME/server_priv/nodes file using the <b>np</b> attribute (see <a href="#">Server node file configuration on page 2572</a>). The virtual processor can relate to a physical core on the node or it can be interpreted as an "execution slot" such as on sites that set the node np value greater than the number of physical cores (or hyper-thread contexts). The ppn value is a characteristic of the hardware, system, and site, and its value is to be determined by the administrator.</li> <li>• <b>gpus=#</b> - Specify the number of GPUs per node requested for this job. The number of GPUs available on a node can be configured in the \$TORQUE_HOME/server_priv/nodes file using the <b>gpu</b> attribute (see <a href="#">Server node file configuration on page 2572</a>). The GPU value is a characteristic of the hardware, system, and site, and its value is to be determined by the administrator.</li> <li>• <b>property</b> - A string assigned by the system administrator specifying a node's features. Check with your administrator as to the node names and properties available to you.</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> TORQUE does not have a TPN (tasks per node) property. You can specify TPN in Moab Workload Manager with TORQUE as your resource manager, but TORQUE does not recognize the property when it is submitted directly to it via <code>qsub</code>.</p> </div> <p>See <a href="#">qsub -l nodes on page 2592</a> for examples.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> By default, the node resource is mapped to a virtual node (that is, directly to a processor, not a full physical compute node). This behavior can be changed within Maui or Moab by setting the <code>JOBNODEMATCHPOLICY</code> parameter. See <a href="#">Appendix A: Moab Parameters on page 923</a> for more information.</p> </div>
<b>opsys</b>	string	Specifies the administrator defined operating system as defined in the MOM configuration file.

Resource	Format	Description
<b>other</b>	string	Allows a user to specify site specific information. This resource is provided for use by the site's scheduling policy. The allowable values and effect on job placement is site dependent.   This does not work for <code>msub</code> using Moab and Maui.
<b>pccput</b>	seconds, or [[HH:]MM:]SS	Maximum amount of CPU time used by any single process in the job.
<b>pmem</b>	<a href="#">size</a>	Maximum amount of physical memory used by any single process of the job. (Ignored on Fujitsu. Not implemented on Digital Unix and HPUNIX.)
<b>procs</b>	procs=<integer>	<i>(Applicable in version 2.5.0 and later.)</i> The number of processors to be allocated to a job. The processors can come from one or more qualified node(s). Only one procs declaration may be used per submitted <a href="#">qsub</a> command.  > qsub -l nodes=3 -l procs=2
<b>procs_bitmap</b>	string	A string made up of 1's and 0's in reverse order of the processor cores requested. A <b>procs_bitmap=1110</b> means the job requests a node that has four available cores, but the job runs exclusively on cores two, three, and four. With this bitmap, core one is not used.  For more information, see <a href="#">Scheduling cores on page 2624</a> .
<b>prologue</b>	string	Specifies a user owned prologue script which will be run after the system prologue and prologue.user scripts at the beginning of a job. The syntax is <code>prologue=&lt;file&gt;</code> . The file can be designated with an absolute or relative path.  For more information, see <a href="#">Appendix G: Prologue and epilogue scripts on page 2817</a> .
<b>pvmem</b>	<a href="#">size</a>	Maximum amount of virtual memory used by any single process in the job. (Ignored on Unicos.)
<b>size</b>	integer	For TORQUE, this resource has no meaning. It is passed on to the scheduler for interpretation. In the Moab scheduler, the size resource is intended for use in Cray installations only.
<b>software</b>	string	Allows a user to specify software required by the job. This is useful if certain software packages are only available on certain systems in the site. This resource is provided for use by the site's scheduling policy. The allowable values and effect on job placement is site dependent. (See <a href="#">License Management on page 663</a> for more information.)

Resource	Format	Description
<b>vmem</b>	<a href="#">size</a>	Maximum amount of virtual memory used by all concurrent processes in the job. (Ignored on Unicors.)
<b>walltime</b>	seconds, or [[HH:]MM:]SS	Maximum amount of real time during which the job can be in the running state.

### size

The size format specifies the maximum amount in terms of bytes or words. It is expressed in the form *integer[suffix]*. The suffix is a multiplier defined in the following table ("b" means bytes [the default] and "w" means words). The size of a word is calculated on the execution server as its word size.

Suffix		Multiplier
b	w	1
kb	kw	1024
mb	mw	1,048,576
gb	gw	1,073,741,824
tb	tw	1,099,511,627,776

### Example 4-10: `qsub -l nodes`

Usage	Description
<code>&gt; qsub -l nodes=12</code>	Request 12 nodes of any type
<code>&gt; qsub -l nodes=2:server+14</code>	Request 2 "server" nodes and 14 other nodes (a total of 16) - this specifies two node_specs, "2:server" and "14"
<code>&gt; qsub -l nodes=s=server:hippi+10:noserver+3:bigmem:hippi</code>	Request (a) 1 node that is a "server" and has a "hippi" interface, (b) 10 nodes that are not servers, and (c) 3 nodes that have a large amount of memory and have hipp
<code>&gt; qsub -l nodes=b2005+b1803+b1813</code>	Request 3 specific nodes by hostname

Usage	Description
> qsub -l nodes=4:ppn=2	Request 2 processors on each of four nodes
> qsub -l nodes=1:ppn=4	Request 4 processors on one node
> qsub -l nodes=2:blue:ppn=2+red:ppn=3+b1014	Request 2 processors on each of two blue nodes, three processors on one red node, and the compute node "b1014"

**Example 4-11:**

This job requests a node with 200MB of available memory:

```
> qsub -l mem=200mb /home/user/script.sh
```

**Example 4-12:**

This job will wait until node01 is free with 200MB of available memory:

```
> qsub -l nodes=node01,mem=200mb /home/user/script.sh
```

## Related topics

- [Job submission on page 2583](#)

## Requesting generic resources

When **generic** resources have been assigned to nodes using the server's nodes file, these resources can be requested at the time of job submission using the *other* field. (See [Managing Shared Cluster Resources \(Floating Resources\) on page 567](#) for details on configuration within Moab).

**Example 4-13: Generic**

This job will run on any node that has the generic resource **matlab**.

```
> qsub -l other=matlab /home/user/script.sh
```



This can also be requested at the time of job submission using the `-W x=GRES:matlab` flag.

## Related topics

- [Requesting resources on page 2586](#)
- [Job submission on page 2583](#)

## Requesting floating resources

When **floating** resources have been set up inside Moab, they can be requested in the same way as **generic** resources. Moab will automatically understand that these resources are floating and will

schedule the job accordingly. (See [Managing Shared Cluster Resources \(Floating Resources\)](#) on page 567 for details on configuration within Moab.)

#### Example 4-14: Floating

This job will run on any node when there are enough floating resources available.

```
> qsub -l other=matlab /home/user/script.sh
```

 This can also be requested at the time of job submission using the `-W x=GRES:matlab` flag.

#### Related topics

- [Requesting resources on page 2586](#)
- [Job submission on page 2583](#)

## Requesting other resources

Many other resources can be requested at the time of job submission using the Moab Workload Manager. See [Resource Manager Extensions on page 618](#) for a list of these supported requests and correct syntax.

#### Related topics

- [Requesting resources on page 2586](#)
- [Job submission on page 2583](#)

## Exported batch environment variables

When a batch job is started, a number of variables are introduced into the job's environment that can be used by the batch script in making decisions, creating output files, and so forth. These variables are listed in the following table:

Variable	Description
<b>PBS_JOBNAME</b>	User specified jobname
<b>PBS_ARRAYID</b>	Zero-based value of job array index for this job (in version 2.2.0 and later)
<b>PBS_GPUFILE</b>	Line-delimited list of GPUs allocated to the job located in <code>\$TORQUE_HOME/aux/jobidgpu</code> . Each line follows the following format: <code>&lt;host&gt;-gpu&lt;number&gt;</code> For example, <code>myhost-gpu1</code> .
<b>PBS_O_WORKDIR</b>	Job's submission directory

Variable	Description
<b>PBS_ENVIRONMENT</b>	N/A
<b>PBS_TASKNUM</b>	Number of tasks requested
<b>PBS_O_HOME</b>	Home directory of submitting user
<b>PBS_MOMPORT</b>	Active port for MOM daemon
<b>PBS_O_LOGNAME</b>	Name of submitting user
<b>PBS_O_LANG</b>	Language variable for job
<b>PBS_JOBCOOKIE</b>	Job cookie
<b>PBS_JOBID</b>	Unique pbs job id
<b>PBS_NODENUM</b>	Node offset number
<b>PBS_NUM_NODES</b>	Number of nodes allocated to the job
<b>PBS_NUM_PPN</b>	Number of procs per node allocated to the job
<b>PBS_O_SHELL</b>	Script shell
<b>PBS_O_HOST</b>	Host on which job script is currently running
<b>PBS_QUEUE</b>	Job queue
<b>PBS_NODEFILE</b>	File containing line delimited list of nodes allocated to the job
<b>PBS_NP</b>	Number of execution slots (cores) for the job
<b>PBS_O_PATH</b>	Path variable used to locate executables within job script

## Related topics

- [Requesting resources on page 2586](#)
- [Job submission on page 2583](#)

## Enabling trusted submit hosts

By default, only the node running the `pbs_server` daemon is allowed to submit jobs. Additional nodes can be trusted as submit hosts by taking any of the following steps:

- Set the `allow_node_submit` server parameter (see [Allowing job submission from compute hosts on page 2568](#)).  
Allows any host trusted as a compute host to also be trusted as a submit host.
- Set the `submit_hosts` server parameter (see [Using the "submit\\_hosts" service parameter on page 2567](#)).  
Allows specified hosts to be trusted as a submit host.
- Use `.rhosts` to enable `ruserok()` based authentication (see [Using RCmd authentication on page 2566](#)).

See [Configuring job submission hosts on page 2566](#) for more information.

**i** When you enable `allow_node_submit` on page 2767, you must also enable the `allow_proxy_user` on page 2767 parameter to allow user proxying when submitting and running jobs.

## Related topics

- [Job submission on page 2583](#)

## Example submit scripts

The following is an example job test script:

```
#!/bin/sh
#
#This is an example script example.sh
#
#These commands set up the Grid Environment for your job:
#PBS -N ExampleJob
#PBS -l nodes=1,walltime=00:01:00
#PBS -q np_workq
#PBS -M YOURUNIQNAME@umich.edu
#PBS -m abe

#print the time and date
date

#wait 10 seconds
sleep 10

#print the time and date again
date
```

## Related topics

- [Job submission on page 2583](#)

## Job files

TORQUE 4.5.0 was updated to accept XML-based job files in addition to the binary job files. The change allows job files to be more human-readable and easier to parse. Below is a sample job file in the new XML format:

```

<?xml version="1.0"?>
<job>
  <version>131842</version>
  <state>1</state>
  <substate>10</substate>
  <server_flags>33</server_flags>
  <start_time>0</start_time>
  <jobid>340</jobid>
  <fileprefix>340</fileprefix>
  <queue>batch</queue>
  <destination_queue></destination_queue>
  <record_type>1</record_type>
  <mom_address>0</mom_address>
  <mom_port>11</mom_port>
  <mom_rmpport>0</mom_rmpport>
  <attributes>
    <Job_Name flags="1">job2.sh</Job_Name>
    <Job_Owner flags="1">echan@moabServer.cn</Job_Owner>
    <job_state flags="1">Q</job_state>
    <queue flags="3">batch</queue>
    <server flags="1">company.com</server>
    <Checkpoint flags="1">u</Checkpoint>
    <ctime flags="1">1384292754</ctime>
    <Error_Path flags="1">moabServer.cn:/home/echan/work/job2.sh.e340</Error_Path>
    <Hold_Types flags="1">n</Hold_Types>
    <Join_Path flags="1">n</Join_Path>
    <Keep_Files flags="1">n</Keep_Files>
    <Mail_Points flags="1">a</Mail_Points>
    <mtime flags="1">1384292754</mtime>
    <Output_Path flags="1">moabServer.cn:/home/echan/work/job2.sh.o340</Output_Path>
    <Priority flags="1">0</Priority>
    <qtime flags="1">1384292754</qtime>
    <Rerunable flags="1">True</Rerunable>
    <Resource_List>
      <epilogue flags="1">/tmp/epilogue.sh</epilogue>
      <neednodes flags="1">moabServer:ppn=1</neednodes>
      <nodect flags="1">1</nodect>
      <nodes flags="1">moabServer:ppn=1</nodes>
    </Resource_List>
    <substate flags="1">10</substate>
    <Variable_List flags="1">PBS_O_QUEUE=batch
PBS_O_HOME=/home/echan
PBS_O_LOGNAME=echan
PBS_O_
PATH=/home/echan/eclipse:/usr/lib/lightdm/lightdm:/usr/local/sbin:/usr/local/bin:/usr/
sbin:/usr/bin:/sbin:/bin:/usr/games:/opt/moab/bin:/opt/moab/sbin
PBS_O_SHELL=/bin/bash
PBS_O_LANG=en_US
PBS_O_WORKDIR=/home/echan/work
PBS_O_HOST=moabServer.cn
PBS_O_SERVER=moabServer
</Variable_List>
    <euser flags="1">echan</euser>
    <egroup flags="5">company</egroup>
    <hop_count flags="1">1</hop_count>
    <queue_rank flags="1">2</queue_rank>
    <queue_type flags="1">E</queue_type>
    <etime flags="1">1384292754</etime>
    <submit_args flags="1">-l nodes=lei:ppn=1 -l epilogue=/tmp/epilogue.sh
./job2.sh</submit_args>
    <fault_tolerant flags="1">False</fault_tolerant>
    <job_radix flags="1">0</job_radix>
  </attributes>
</job>

```

```
<submit_host flags="1">lei.ac</submit_host>
</attributes>
</job>
```

The above job was submitted with this submit command:

```
qsub -l nodes=moabServer:ppn=1 -l epilogue=/tmp/epilogue.sh ./job2.sh
```

Related topics

- [Job submission on page 2583](#)

## Monitoring jobs

TORQUE allows users and administrators to monitor submitted jobs with the [qstat](#) command. If the command is run by a non-administrative user, it will output just that user's jobs. For example:

```
> qstat
Job id      Name      User      Time Use S Queue
-----
4807       scatter  user01    12:56:34 R batch
...
```

Related topics

- [Submitting and managing jobs on page 2582](#)

## Canceling jobs

TORQUE allows users and administrators to cancel submitted jobs with the [qdel](#) command. The job will be sent TERM and KILL signals killing the running processes. When the top-level job script exits, the job will exit. The only parameter is the ID of the job to be canceled.

If a job is canceled by an operator or manager, an email notification will be sent to the user. Operators and managers may add a comment to this email with the `-m` option.

```
$ qstat
Job id      Name      User      Time Use S Queue
-----
4807       scatter  user01    12:56:34 R batch
...
$ qdel -m "hey! Stop abusing the NFS servers" 4807
$
```

Related topics

- [Submitting and managing jobs on page 2582](#)

## Job preemption

TORQUE supports job preemption by allowing authorized users to suspend and resume jobs. This is supported using one of two methods. If the node supports OS-level preemption, TORQUE will recognize that during the configure process and enable it. Otherwise, the MOM may be configured to launch a custom *checkpoint script* in order to support preempting a job. Using a custom checkpoint script requires that the job understand how to resume itself from a checkpoint after the preemption occurs.

### Configuring a checkpoint script on a MOM

To configure the MOM to support a checkpoint script, the `$checkpoint_script` parameter must be set in the MOM's configuration file found in `TORQUE_HOME/mom_priv/config`. The checkpoint script should have execute permissions set. A typical configuration file might look as follows:

mom\_priv/config:

```
$pbsserver      node06
$logevent       255
$restricted     *.mycluster.org
$checkpoint_script /opt/moab/tools/mom-checkpoint.sh
```

The second thing that must be done to enable the checkpoint script is to change the value of `MOM_CHECKPOINT` to **1** in `/src/include/pbs_config.h`. (In some instances, `MOM_CHECKPOINT` may already be defined as 1.) The new line should be as follows:

/src/include/pbs\_config.h:

```
#define MOM_CHECKPOINT 1
```

#### Related topics

- [Submitting and managing jobs on page 2582](#)

## Keeping completed jobs

TORQUE provides the ability to report on the status of completed jobs for a configurable duration after the job has completed. This can be enabled by setting the [keep\\_completed on page 2631](#) attribute on the job execution queue or the [keep\\_completed on page 2772](#) parameter on the server. This should be set to the number of seconds that jobs should be held in the queue. If you set `keep_completed` on the job execution queue, completed jobs will be reported in the **C** state and the exit status is seen in the `exit_status` job attribute.

- i** If the Mother Superior and TORQUE server are on the same server, expect the following behavior:
- When `keep_completed` is set, the job spool files will be deleted when the specified time arrives and TORQUE purges the job from memory.
  - When `keep_completed` is not set, TORQUE deletes the job spool files upon job completion.
  - If you manually purge a job (`qdel -p`) before the job completes or time runs out, TORQUE will never delete the spool files.

By maintaining status information about completed (or canceled, failed, etc.) jobs, administrators can better track failures and improve system performance. This allows TORQUE to better communicate with Moab Workload Manager and track the status of jobs. This gives Moab the ability to track specific failures and to schedule the workload around possible hazards. (See [NODEFAILURERESERVETIME](#) for more information.)

#### Related topics

- [Submitting and managing jobs on page 2582](#)

## Job checkpoint and restart

While TORQUE has had a job checkpoint and restart capability for many years, this was tied to machine specific features. Now TORQUE supports BLCR—an architecture independent package that provides for process checkpoint and restart.

- i** The support for BLCR is only for serial jobs, not for any MPI type jobs.

This section contains these topics:

- [Introduction to BLCR on page 2601](#)
- [Configuration files and scripts on page 2602](#)
- [Starting a checkpointable job on page 2609](#)
- [Checkpointing a job on page 2610](#)
- [Restarting a job on page 2610](#)
- [Acceptance tests on page 2611](#)

#### Related topics

- [Submitting and managing jobs on page 2582](#)

## Introduction to BLCR

BLCR is a kernel level package. It must be downloaded and installed from [BLCR](#).

After building and making the package, it must be installed into the kernel with commands as follows. These can be installed into the file `/etc/modules` but all of the testing was done with explicit invocations of **modprobe**.

Installing BLCR into the kernel:

```
# /sbin/insmod /usr/local/lib/blcr/2.6.12-1.234/blcr_imports.ko
# /sbin/insmod /usr/local/lib/blcr/2.6.12-1.234/blcr_vmadump.ko
# /sbin/insmod /usr/local/lib/blcr/2.6.12-1.234/blcr.ko
```

The BLCR system provides four command line utilities:

- `cr_checkpoint`
- `cr_info`
- `cr_restart`
- `cr_run`

For more information about BLCR, see the [BLCR Administrator's Guide](#).

Related topics

- [Job checkpoint and restart on page 2601](#)

## Configuration files and scripts

Configuring and Building TORQUE for BLCR:

```
> ./configure --enable-unixsockets=no --enable-blcr
> make
> sudo make install
```

Depending on where BLCR is installed you may also need to use the following configure options to specify BLCR paths:

Option	Description
<code>--with-blcr-include=DIR</code>	include path for <code>libcr.h</code>
<code>--with-blcr-lib=DIR</code>	lib path for <code>libcr</code>
<code>--with-blcr-bin=DIR</code>	bin path for BLCR utilities

The `pbs_mom` configuration file located in `/var/spool/torque/mom_priv` must be modified to identify the script names associated with invoking the BLCR commands. The following variables should be used in the configuration file when using BLCR checkpointing.

Variable	Description
<code>\$checkpoint_interval</code>	How often periodic job checkpoints will be taken (minutes)
<code>\$checkpoint_script</code>	The name of the script file to execute to perform a job checkpoint
<code>\$restart_script</code>	The name of the script file to execute to perform a job restart
<code>\$checkpoint_run_exe</code>	The name of an executable program to be run when starting a checkpointable job (for BLCR, <code>cr_run</code> )

The following example shows the contents of the configuration file used for testing the BLCR feature in TORQUE.

 The script files below must be executable by the user. Be sure to use `chmod` to set the permissions to 754.

#### Example 4-15: Script file permissions

```
# chmod 754 blcr*
# ls -l
total 20
-rwxr-xr-- 1 root root 2112 2008-03-11 13:14 blcr_checkpoint_script
-rwxr-xr-- 1 root root 1987 2008-03-11 13:14 blcr_restart_script
-rw-r--r-- 1 root root 215 2008-03-11 13:13 config
drwxr-x--x 2 root root 4096 2008-03-11 13:21 jobs
-rw-r--r-- 1 root root 7 2008-03-11 13:15 mom.lock
```

#### Example 4-16: `mom_priv/config`

```
$checkpoint_script /var/spool/torque/mom_priv/blcr_checkpoint_script
$restart_script /var/spool/torque/mom_priv/blcr_restart_script
$checkpoint_run_exe /usr/local/bin/cr_run
$pbsserver makuu.cridomain
$loglevel 7
```

*Example 4-17: mom\_priv/blcr\_checkpoint\_script*

```

#!/usr/bin/perl
#####
#
# Usage: checkpoint_script
#
# This script is invoked by pbs_mom to checkpoint a job.
#
#####
use strict;
use Sys::Syslog;

# Log levels:
# 0 = none -- no logging
# 1 = fail -- log only failures
# 2 = info -- log invocations
# 3 = debug -- log all subcommands
my $logLevel = 3;

logPrint(2, "Invoked: $0 " . join(' ', @ARGV) . "\n");

my ($sessionId, $jobId, $userId, $signalNum, $checkpointDir, $checkpointName);
my $usage =
    "Usage: $0          \n";

# Note that depth is not used in this script but could control a limit to the number
of checkpoint
# image files that are preserved on the disk.
#
# Note also that a request was made to identify whether this script was invoked by the
job's
# owner or by a system administrator. While this information is known to pbs_server,
it
# is not propagated to pbs_mom and thus it is not possible to pass this to the script.

# Therefore, a workaround is to invoke qmgr and attempt to set a trivial variable.
# This will fail if the invoker is not a manager.

if (@ARGV == 7)
{
    ($sessionId, $jobId, $userId, $checkpointDir, $checkpointName, $signalNum $depth)
=
    @ARGV;
}
else { logDie(1, $usage); }

# Change to the checkpoint directory where we want the checkpoint to be created
chdir $checkpointDir
    or logDie(1, "Unable to cd to checkpoint dir ($checkpointDir): ${!}\n")
    if $logLevel;

my $cmd = "cr_checkpoint";
$cmd .= " --signal $signalNum" if $signalNum;
$cmd .= " --tree $sessionId";
$cmd .= " --file $checkpointName";
my $output = `$cmd 2>&1`;
my $rc = $? >> 8;
logDie(1, "Subcommand ($cmd) failed with rc=$rc:\n$output")
    if $rc && $logLevel >= 1;
logPrint(3, "Subcommand ($cmd) yielded rc=$rc:\n$output")
    if $logLevel >= 3;
exit 0;

#####
# logPrint($message)
# Write a message (to syslog) and die
#####
sub logPrint
{

```

```
my ($level, $message) = @_;  
my @severity = ('none', 'warning', 'info', 'debug');  
  
return if $level > $logLevel;  
  
openlog('checkpoint_script', '', 'user');  
syslog($severity[$level], $message);  
closelog();  
}  
  
#####  
# logDie($message)  
# Write a message (to syslog) and die  
#####  
sub logDie  
{  
    my ($level, $message) = @_;  
    logPrint($level, $message);  
    die($message);  
}
```

*Example 4-18: mom\_priv/blcr\_restart\_script*

```

#!/usr/bin/perl
#####
#
# Usage: restart_script
#
# This script is invoked by pbs_mom to restart a job.
#
#####
use strict;
use Sys::Syslog;

# Log levels:
# 0 = none -- no logging
# 1 = fail -- log only failures
# 2 = info -- log invocations
# 3 = debug -- log all subcommands
my $logLevel = 3;

logPrint(2, "Invoked: $0 " . join(' ', @ARGV) . "\n");

my ($sessionId, $jobId, $userId, $checkpointDir, $restartName);
my $usage =
"Usage: $0 \n";
if (@ARGV == 5)
{
    ($sessionId, $jobId, $userId, $checkpointDir, $restartName) =
        @ARGV;
}
else { logDie(1, $usage); }

# Change to the checkpoint directory where we want the checkpoint to be created
chdir $checkpointDir
    or logDie(1, "Unable to cd to checkpoint dir ($checkpointDir): $!\n")
    if $logLevel;

my $cmd = "cr_restart";
$cmd .= " $restartName";
my $output = ` $cmd 2>&1 `;
my $rc = $? >> 8;
logDie(1, "Subcommand ($cmd) failed with rc=$rc:\n$output")
    if $rc && $logLevel >= 1;
logPrint(3, "Subcommand ($cmd) yielded rc=$rc:\n$output")
    if $logLevel >= 3;
exit 0;

#####
# logPrint($message)
# Write a message (to syslog) and die
#####
sub logPrint
{
    my ($level, $message) = @_ ;
    my @severity = ('none', 'warning', 'info', 'debug');

    return if $level > $logLevel;
    openlog('restart_script', '', 'user');
    syslog($severity[$level], $message);
    closelog();
}

#####
# logDie($message)
# Write a message (to syslog) and die
#####
sub logDie
{
    my ($level, $message) = @_ ;

    logPrint($level, $message);
}

```

```
die($message);
}
```

## Related topics

- [Job checkpoint and restart on page 2601](#)

## Starting a checkpointable job

Not every job is checkpointable. A job for which checkpointing is desirable must be started with the `-c` command line option. This option takes a comma-separated list of arguments that are used to control checkpointing behavior. The list of valid options available in the 2.4 version of TORQUE is show below.

Option	Description
<b>none</b>	No checkpointing (not highly useful, but included for completeness).
<b>enabled</b>	Specify that checkpointing is allowed, but must be explicitly invoked by either the <code>qhold</code> or <code>qchkpt</code> commands.
<b>shutdown</b>	Specify that checkpointing is to be done on a job at <code>pbs_mom</code> shutdown.
<b>periodic</b>	Specify that periodic checkpointing is enabled. The default interval is 10 minutes and can be changed by the <code>\$checkpoint_interval</code> option in the MOM configuration file, or by specifying an interval when the job is submitted.
<b>interval=minutes</b>	Specify the checkpoint interval in minutes.
<b>depth=number</b>	Specify a number (depth) of checkpoint images to be kept in the checkpoint directory.
<b>dir=path</b>	Specify a checkpoint directory (default is <code>/var/spool/torque/checkpoint</code> ).

### Example 4-19: Sample test program

```
#include "stdio.h"
int main( int argc, char *argv[] )
{
  int i;
  for (i=0; i<100; i++)
  {
    printf("i = %d\n", i);
    fflush(stdout);
    sleep(1);
  }
}
```

### Example 4-20: Instructions for building test program

```
> gcc -o test test.c
```

**Example 4-21: Sample test script**

```
#!/bin/bash ./test
```

**Example 4-22: Starting the test job**

```
> qstat
> qsub -c enabled,periodic,shutdown,interval=1 test.sh
77.jakaa.cridomain
> qstat
Job id          Name          User          Time Use S Queue
-----
77.jakaa       test.sh       jsmith        0 Q batch
>
```

If you have no scheduler running, you might need to start the job with [grun](#).

As this program runs, it writes its output to a file in `/var/spool/torque/spool`. This file can be observed with the command `tail -f`.

**Related topics**

- [Job checkpoint and restart on page 2601](#)

## Checkpointing a job

Jobs are checkpointed by issuing a [ghold](#) command. This causes an image file representing the state of the process to be written to disk. The directory by default is `/var/spool/torque/checkpoint`.

This default can be altered at the queue level with the `qmgr` command. For example, the command `qmgr -c set queue batch checkpoint_dir=/tmp` would change the checkpoint directory to `/tmp` for the queue 'batch'.

The default directory can also be altered at job submission time with the `-c dir=/tmp` command line option.

The name of the checkpoint directory and the name of the checkpoint image file become attributes of the job and can be observed with the command `qstat -f`. Notice in the output the names **checkpoint\_dir** and **checkpoint\_name**. The variable `checkpoint_name` is set when the image file is created and will not exist if no checkpoint has been taken.

A job can also be checkpointed without stopping or holding the job with the command [gchkpt](#).

**Related topics**

- [Job checkpoint and restart on page 2601](#)

## Restarting a job

### Restarting a job in the Held state

The [grls](#) command is used to restart the hibernated job. If you were using the `tail -f` command to watch the output file, you will see the test program start counting again.

It is possible to use the [galter](#) command to change the name of the checkpoint file associated with a job. This could be useful if there were several job checkpoints and it restarting the job from an older image was specified.

## Restarting a job in the Completed state

In this case, the job must be moved to the Queued state with the [grerun](#) command. Then the job must go to the Run state either by action of the scheduler or if there is no scheduler, through using the [grun](#) command.

Related topics

- [Job checkpoint and restart on page 2601](#)

## Acceptance tests

A number of tests were made to verify the functioning of the BLCR implementation. See [Appendix M: BLCR acceptance tests on page 2836](#) for a description of the testing.

Related topics

- [Job checkpoint and restart on page 2601](#)

## Job exit status

Once a job under TORQUE has completed, the `exit_status` attribute will contain the result code returned by the job script. This attribute can be seen by submitting a `qstat -f` command to show the entire set of information associated with a job. The `exit_status` field is found near the bottom of the set of output lines.

**Example 4-23: qstat -f (job failure)**

```

Job Id: 179.host
Job_Name = STDIN
Job_Owner = user@host
job_state = C
queue = batchq server = host
Checkpoint = u ctime = Fri Aug 29 14:55:55 2008
Error_Path = host:/opt/moab/STDIN.e179
exec_host = node1/0
Hold_Types = n
Join_Path = n
Keep_Files = n
Mail_Points = a
mtime = Fri Aug 29 14:55:55 2008
Output_Path = host:/opt/moab/STDIN.o179
Priority = 0
qtime = Fri Aug 29 14:55:55 2008
Rerunable = True Resource_List.ncpus = 2
Resource_List.nodect = 1
Resource_List.nodes = node1
Variable_List = PBS_O_HOME=/home/user,PBS_O_LOGNAME=user,
PBS_O_PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:,PBS_O_
SHELL=/bin/bash,PBS_O_HOST=host,
PBS_O_WORKDIR=/opt/moab,PBS_O_QUEUE=batchq
sched_hint = Post job file processing error; job 179.host on host node1/0Ba
d UID for job execution REJHOST=pala.cridomain MSG=cannot find user 'user' in
password file
etime = Fri Aug 29 14:55:55 2008
exit_status = -1
    
```

**i** The value of `Resource_List.*` is the amount of resources requested.

This code can be useful in diagnosing problems with jobs that may have unexpectedly terminated.

If TORQUE was unable to start the job, this field will contain a negative number produced by the `pbs_mom`. Otherwise, if the job script was successfully started, the value in this field will be the return value of the script.

**Example 4-24: TORQUE supplied exit codes**

Name	Value	Description
<b>JOB_EXEC_OK</b>	0	Job execution successful
<b>JOB_EXEC_FAIL1</b>	-1	Job execution failed, before files, no retry
<b>JOB_EXEC_FAIL2</b>	-2	Job execution failed, after files, no retry
<b>JOB_EXEC_RETRY</b>	-3	Job execution failed, do retry
<b>JOB_EXEC_INITABT</b>	-4	Job aborted on MOM initialization
<b>JOB_EXEC_INITRST</b>	-5	Job aborted on MOM init, chkpt, no migrate

Name	Value	Description
<b>JOB_EXEC_INITRMG</b>	-6	Job aborted on MOM init, chkpt, ok migrate
<b>JOB_EXEC_BADRESRT</b>	-7	Job restart failed
<b>JOB_EXEC_CMDFAIL</b>	-8	Exec() of user command failed
<b>JOB_EXEC_STDOUTFAIL</b>	-9	Could not create/open stdout stderr files
<b>JOB_EXEC_OVERLIMIT_MEM</b>	-10	Job exceeded a memory limit
<b>JOB_EXEC_OVERLIMIT_WT</b>	-11	Job exceeded a walltime limit
<b>JOB_EXEC_OVERLIMIT_CPUT</b>	-12	Job exceeded a CPU time limit

## Example 4-25: Exit code from C program

```

$ cat error.c
#include
#include

int
main(int argc, char *argv)
{
    exit(256+11);
}

$ gcc -o error error.c

$ echo ./error | qsub
180.xxx.yyy

$ qstat -f
Job Id: 180.xxx.yyy
  Job_Name = STDIN
  Job_Owner = test.xxx.yyy
  resources_used.cput = 00:00:00
  resources_used.mem = 0kb
  resources_used.vmem = 0kb
  resources_used.walltime = 00:00:00
  job_state = C
  queue = batch
  server = xxx.yyy
  Checkpoint = u
  ctime = Wed Apr 30 11:29:37 2008
  Error_Path = xxx.yyy:/home/test/STDIN.e180
  exec_host = node01/0
  Hold_Types = n
  Join_Path = n
  Keep_Files = n
  Mail_Points = a
  mtime = Wed Apr 30 11:29:37 2008
  Output_Path = xxx.yyy:/home/test/STDIN.o180
  Priority = 0
  qtime = Wed Apr 30 11:29:37 2008
  Rerunnable = True
  Resource_List.needsnodes = 1
  Resource_List.nodect = 1
  Resource_List.nodes = 1
  Resource_List.walltime = 01:00:00
  session_id = 14107
  substate = 59
  Variable_List = PBS_O_HOME=/home/test,PBS_O_LANG=en_US.UTF-8,
    PBS_O_LOGNAME=test,
    PBS_O_PATH=/usr/local/perltests/bin:/home/test/bin:/usr/local/s
    bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games,
    PBS_O_SHELL=/bin/bash,PBS_SERVER=xxx.yyy,
    PBS_O_HOST=xxx.yyy,PBS_O_WORKDIR=/home/test,
    PBS_O_QUEUE=batch
  euser = test
  egroup = test
  hashname = 180.xxx.yyy
  queue_rank = 8
  queue_type = E
  comment = Job started on Wed Apr 30 at 11:29
  etime = Wed Apr 30 11:29:37 2008
  exit_status = 11
  start_time = Wed Apr 30 11:29:37 2008
  start_count = 1

```

Notice that the C routine `exit` passes only the low order byte of its argument. In this case, `256+11` is really `267` but the resulting exit code is only `11` as seen in the output.

Related topics

- [Job checkpoint and restart on page 2601](#)
- [Submitting and managing jobs on page 2582](#)

## Service jobs

TORQUE service jobs are a special kind of job that is treated differently by TORQUE than normal batch jobs. TORQUE service jobs are *not* related to Moab's dynamic service jobs. A TORQUE service job cannot dynamically grow and shrink in size over time.

Jobs are marked as service jobs at the time they are submitted to Moab or TORQUE. Just like a normal job, a script file is specified with the job. In a batch job, the contents of the script file are taken by TORQUE and executed on the compute nodes. For a service job, however, the script file is assumed to respond to certain command-line arguments. Instead of just executing the script, TORQUE will use these command-line arguments to start, stop, and check on the status of the job. Listed below are the three command-line arguments that must be supported by any script submitted as part of a TORQUE service job:

- `start`: The script should take this argument and launch its service/workload. The script should remain executing/running until the service stops.
- `stop`: The script should take this argument and stop the service/workload that was earlier started.
- `status`: The script should take this argument and return, via standard out, either "running" if the service/workload is running as expected or "stopped" if the service is not running.

This feature was created with long-running services in mind. The command-line arguments should be familiar to users who interact with Unix services, as each of the service scripts found in `/etc/init.d/` also accept and respond to the arguments as explained above.

For example, if a user wants to start the Apache 2 server on a compute node, they can use a TORQUE service job and specify a script which will start, stop, and check on the status of the "httpd" daemon--possibly by using the already present `/etc/init.d/httpd` script.



If you wish to submit service jobs only through TORQUE, no special version of Moab is required. If you wish to submit service jobs using Moab's `msub`, then Moab 5.4 is required.

For details, see these topics:

- [Submitting service jobs on page 2616](#)
- [Submitting service jobs in MCM on page 2616](#)
- [Managing service jobs on page 2616](#)

## Submitting service jobs

There is a new option to `qsub`, "-s" which can take either a 'y' or 'n' (yes or no, respectively). When "-s y" is present, then the job is marked as a service job.

```
qsub -l walltime=100:00:00,nodes=1 -s y service_job.py
```

The example above submits a job to TORQUE with a walltime of 100 hours, one node, and it is marked as a service job. The script "service\_job.py" will be used to start, stop, and check the status of the service/workload started on the compute nodes.

Moab, as of version 5.4, is able to accept the "-s y" option when `mbsub` is used for submission. Moab will then pass this information to TORQUE when the job is migrated.

### Related topics

- [Service jobs on page 2615](#)

## Submitting service jobs in MCM

Submitting a service job in MCM requires the latest Adaptive Computing Suite snapshot of MCM. It also requires MCM to be started with the "`--future=2`" option.

Once MCM is started, open the **Create Workload** window and verify **Show Advanced Options** is checked. Notice that there is a **Service** checkbox that can be selected in the **Flags/Options** area. Use this to specify the job is a service job.

### Related topics

- [Service jobs on page 2615](#)

## Managing service jobs

Managing a service job is done much like any other job; only a few differences exist.

Examining the job with `qstat -f` will reveal that the job has the `service = True` attribute. Non-service jobs will not make any mention of the "service" attribute.

Canceling a service job is done with `qdel`, `mjobctl -c`, or through any of the GUI's as with any other job. TORQUE, however, cancels the job by calling the service script with the "stop" argument instead of killing it directly. This behavior also occurs if the job runs over its wallclock and TORQUE/Moab is configured to cancel the job.

If a service job completes when the script exits after calling it with "start," or if TORQUE invokes the script with "status" and does not get back "running," it will *not* be terminated by using the "stop" argument.

### Related topics

- [Service jobs on page 2615](#)

# Managing nodes

This section contains information about adding and configuring compute nodes. It explains how to work with host security for systems that require dedicated access to compute nodes. It also contains information about scheduling specific cores on a node at job submission.

For details, see these topics:

- [Adding nodes on page 2617](#)
- [Node properties on page 2618](#)
- [Changing node state on page 2618](#)
- [Host security on page 2621](#)
- [Linux cpuset support on page 2623](#)
- [Scheduling cores on page 2624](#)

## Adding nodes

TORQUE can add and remove nodes either dynamically with [qmgr](#) or by manually editing the `TORQUE_HOME/server_priv/nodes` file (see [Initializing/Configuring TORQUE on the server \(pbs\\_server\) on page 2551](#)).

### Run-time node changes

TORQUE can dynamically add nodes with the `qmgr` command. For example, the following command will add node **node003**:

```
> qmgr -c "create node node003"
```

The above command appends the `$TORQUE_HOME/server_priv/nodes` file with:

```
node003
```

Nodes can also be removed with a similar command:

```
> qmgr -c "delete node node003"
```

**i** Typically, an administrator will want to change the state of a node instead of remove it (for details, see [Changing node state on page 2618](#)).

**i** When you make changes to nodes – whether by using `qmgr` or directly editing the `nodes` file – you must restart `pbs_server` for those changes to take effect.

Related topics

- [Managing nodes on page 2617](#)

## Node properties

TORQUE can associate properties with nodes to aid in identifying groups of nodes. It's typical for a site to conglomerate a heterogeneous set of resources. To identify the different sets, properties can be given to each node in a set. For example, a group of nodes that has a higher speed network connection could have the property "ib". TORQUE can set, update, or remove properties either dynamically with [qmgr](#) or by manually editing the `nodes` file.

### Run-time node changes

TORQUE can dynamically change the properties of a node with the `qmgr` command. For example, note the following to give **node001** the properties of "bigmem" and "dualcore":

```
> qmgr -c "set node node001 properties = bigmem"
> qmgr -c "set node node001 properties += dualcore"
```

To relinquish a stated property, use the "-" operator.

### Manual node changes

The properties of each node are enumerated in `TORQUE_HOME/server_priv/nodes`. The feature(s) must be in a space delimited list after the node name. For example, to give **node001** the properties of "bigmem" and "dualcore" and **node002** the properties of "bigmem" and "matlab," edit the `nodes` file to contain the following:

`server_priv/nodes:`

```
node001 bigmem dualcore
node002 np=4 bigmem matlab
```



For changes to the `nodes` file to be activated, `pbs_server` must be restarted.



For a full description of this file, please see the *PBS Administrator Guide*.

#### Related topics

- [Job submission on page 2583](#)
- [Managing nodes on page 2617](#)

## Changing node state

A common task is to prevent jobs from running on a particular node by marking it *offline* with `pbsnodes -o nodename`. Once a node has been marked offline, the scheduler will no longer consider it available for new jobs. Simply use `pbsnodes -c nodename` when the node is returned to service.

Also useful is `pbsnodes -l`, which lists all nodes with an interesting state, such as down, unknown, or offline. This provides a quick glance at nodes that might be having a problem. (See [pbsnodes](#) for details.)

## Related topics

- [Managing nodes on page 2617](#)

## Changing node power states

In TORQUE 5.0.0 and later, the `pbsnodes -m` command can modify the power state of nodes. Node cannot go from one low-power state to another low-power state. They must be brought up to the Running state and then moved to the new low-power state. The supported power states are:

State	Description
<b>Running</b>	<ul style="list-style-type: none"> <li>• Physical machine is actively working</li> <li>• Power conservation is on a per-device basis</li> <li>• Processor power consumption controlled by P-states</li> </ul>
<b>Standby</b>	<ul style="list-style-type: none"> <li>• System appears off</li> <li>• Processor halted (OS executes a "halt" instruction)</li> <li>• Processor maintains CPU and system cache state</li> <li>• RAM refreshed to maintain memory state</li> <li>• Machine in low-power mode</li> <li>• Requires interrupt to exit state</li> <li>• Lowest-latency sleep state - has no effect on software</li> </ul>
<b>Suspend</b>	<ul style="list-style-type: none"> <li>• System appears off</li> <li>• Processor and support chipset have no power</li> <li>• OS maintains CPU, system cache, and support chipset state in memory</li> <li>• RAM in slow refresh</li> <li>• Machine in lowest-power state</li> <li>• Usually requires specific interrupt (keyboard, mouse) to exit state</li> <li>• Third lowest-latency sleep state - system must restore power to processor and support chipset</li> </ul>
<b>Hibernate</b>	<ul style="list-style-type: none"> <li>• System is off</li> <li>• Physical machine state and memory saved to disk</li> <li>• Requires restoration of power and machine state to exit state</li> <li>• Second highest-latency sleep state - system performs faster boot using saved machine state and copy of memory</li> </ul>
<b>Shutdown</b>	<ul style="list-style-type: none"> <li>• Equivalent to <code>shutdown now</code> command as root</li> </ul>

In order to wake nodes and bring them up to a running state:

- the nodes must support, and be configured to use, Wake-on-LAN (WOL).
- the pbsnodes command must report the node's MAC address correctly.

### To configure nodes to use Wake-on-LAN

1. Enable WOL in the BIOS for each node. If needed, contact your hardware manufacturer for details.
2. Use the `ethtool` command to determine what types of WOL packets your hardware supports. TORQUE uses the `g` packet. If the `g` packet is not listed, you cannot use WOL with TORQUE.

```
[root]# ethtool eth0
Settings for eth0:
    Supported ports: [ TP ]
    Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full

    Supported pause frame use: No
    Supports auto-negotiation: Yes
    Advertised link modes:  10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full

    Advertised pause frame use: No
    Advertised auto-negotiation: Yes
    Speed: 100Mb/s
    Duplex: Full
    Port: Twisted Pair
    PHYAD: 2
    Transceiver: internal
    Auto-negotiation: on
    MDI-X: off
    Supports Wake-on: pumbg
    Wake-on: p
    Current message level: 0x00000007 (7)
                           drv probe link

    Link detected: yes
```

*This Ethernet interface supports the `g` WOL packet, but is currently set to use the `p` packet.*

3. If your Ethernet interface supports the `g` packet, but is configured for a different packet, use `ethtool -s <interface> wol g` to configure it to use `g`.

```

[root]# ethtool -s eth0 wol g
[root]# ethtool eth0
Settings for eth0:
    Supported ports: [ TP ]
    Supported link modes:   10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full

    Supported pause frame use: No
    Supports auto-negotiation: Yes
    Advertised link modes:  10baseT/Half 10baseT/Full
                           100baseT/Half 100baseT/Full
                           1000baseT/Full

    Advertised pause frame use: No
    Advertised auto-negotiation: Yes
    Speed: 100Mb/s
    Duplex: Full
    Port: Twisted Pair
    PHYAD: 2
    Transceiver: internal
    Auto-negotiation: on
    MDI-X: off
    Supports Wake-on: pumbg
    Wake-on: g
    Current message level: 0x00000007 (7)
                           drv probe link

    Link detected: yes

```

Now the power state of your nodes can be modified and they can be woken up from power-saving states.

#### Related topics

- [pbsnodes on page 2705](#)

## Host security

### Enabling PAM with TORQUE

TORQUE is able to take advantage of the authentication services provided through Pluggable Authentication Modules (PAM) to help administrators manage access to compute nodes by users. The PAM module available in TORQUE is located in the PAM security directory. This module, when used in conjunction with other PAM modules, restricts access to the compute node unless the user has a job currently running on the node. The following configurations are examples only. For more information about PAM, see the [PAM \(Pluggable Authentication Modules\) documentation](#) from LinuxDocs.

To enable TORQUE PAM configure TORQUE using the `--with-pam` option. Using `--with-pam` is sufficient but if your PAM security modules are not in the default `/lib/security` or `/lib64/security` directory, you can specify the location using `--with-pam=<DIR>` where `<DIR>` is the directory where you want the modules to be installed. When TORQUE is installed the files `pam_pbssimpleauth.la` and `pam_pbssimpleauth.so` appear in `/lib/security`, `/lib64/security`, or the directory designated on the configuration line.

PAM is very flexible and policies vary greatly from one site to another. The following example restricts users trying to access a node using SSH. Administrators need to assess their own installations and decide how to apply the TORQUE PAM restrictions.

In this example, after installing TORQUE with PAM enabled, you would add the following two lines to `/etc/pam.d/sshd`:

```
account required pam_pbssimpleauth.so
account required pam_access.so
```

In `/etc/security/access.conf` make sure all users who access the compute node are added to the configuration. This is an example which allows the users `root`, `george`, `allen`, and `michael` access.

```
 -:ALL EXCEPT root george allen michael torque:ALL
```

With this configuration, if user `george` has a job currently running on the compute node, `george` can use `ssh` to login to the node. If there are currently no jobs running, `george` is disconnected when attempting to login.

TORQUE PAM is good at keeping users out who do not have jobs running on a compute node. However, it does not have the ability to force a user to log out once they are in. To accomplish this use epilogue or prologue scripts to force users off the system.

## Legacy TORQUE PAM configuration

There is an alternative PAM configuration for TORQUE that has been available since 2006. It can be found in the `contrib/pam_authuser` directory of the source tree. Adaptive Computing does not currently support this method but the instructions are given here for those who are currently using it and for those who wish to use it.

For systems requiring dedicated access to compute nodes (for example, users with sensitive data), TORQUE prologue and epilogue scripts provide a vehicle to leverage the authentication provided by linux-PAM modules. (See [Appendix G: Prologue and epilogue scripts on page 2817](#) for more information.)

### To allow only users with running jobs (and root) to access compute nodes

1. Untar `contrib/pam_authuser.tar.gz` (found in the src tar ball).
2. Compile `pam_authuser.c` with `make` and `make install` on every compute node.
3. Edit `/etc/system-auth` as described in `README.pam_authuser`, again on every compute node.
4. Either make a tarball of the epilogue\* and prologue\* scripts (to preserve the symbolic link) and untar it in the `mom_priv` directory, or just copy epilogue\* and prologue\* to `mom_priv/`.

The prologue\* scripts are Perl scripts that add the user of the job to `/etc/authuser`. The epilogue\* scripts then remove the first occurrence of the user from `/etc/authuser`. File locking is employed in all scripts to eliminate the chance of race conditions. There is also some commented code in the epilogue\* scripts, which, if uncommented, kills all processes owned by the user (using `pkill`), provided that the user doesn't have another valid job on the same node.

[prologue](#) and [epilogue](#) scripts were added to the `pam_authuser` tarball in version 2.1 of TORQUE.

### Related topics

- [Managing nodes on page 2617](#)

## Linux cpuset support

- [Cpuset overview on page 2623](#)
- [Cpuset support on page 2623](#)
- [Cpuset configuration on page 2623](#)
- [Cpuset advantages / disadvantages on page 2624](#)

### Cpuset overview

Linux kernel 2.6 Cpusets are logical, hierarchical groupings of CPUs and units of memory. Once created, individual processes can be placed within a cpuset. The processes will only be allowed to run/access the specified CPUs and memory. Cpusets are managed in a virtual file system mounted at `/dev/cpuset`. New cpusets are created by simply making new directories. Cpusets gain CPUs and memory units by simply writing the unit number to files within the cpuset.

### Cpuset support

**i** All nodes using cpusets must have the `hwloc` library version 1.2 or higher installed.

When started, `pbs_mom` will create an initial top-level cpuset at `/dev/cpuset/torque`. This cpuset contains all CPUs and memory of the host machine. If this "torqueset" already exists, it will be left unchanged to allow the administrator to override the default behavior. All subsequent cpusets are created within the torqueset.

When a job is started, the jobset is created at `/dev/cpuset/torque/$jobid` and populated with the CPUs listed in the `exec_host job` attribute. Also created are individual tasksets for each CPU within the jobset. This happens before prologue, which allows it to be easily modified, and it happens on all nodes.

The top-level batch script process is executed in the jobset. Tasks launched through the TM interface (`pbsdsh` and PW's `mpiexec`) will be executed within the appropriate taskset.

On job exit, all tasksets and the jobset are deleted.

### Cpuset configuration

#### To configure cpuset

1. As root, mount the virtual filesystem for cpusets:

```
mkdir /dev/cpuset
mount -t cpuset none /dev/cpuset
```

**i** Do this for each MOM that is to use cpusets.

2. Because `cpuset` usage is a build-time option in TORQUE, you must add `--enable-cpuset` to your configure options:

```
./configure --enable-cpuset
```

3. Use this configuration for the MOMs across your system.

## Cpuset advantages / disadvantages

Presently, any job can request a single CPU and proceed to use everything available in the machine. This is occasionally done to circumvent policy, but most often is simply an error on the part of the user. Cpuset support will easily constrain the processes to not interfere with other jobs.

Jobs on larger NUMA systems may see a performance boost if jobs can be intelligently assigned to specific CPUs. Jobs may perform better if striped across physical processors, or contained within the fewest number of memory controllers.

TM tasks are constrained to a single core, thus a multi-threaded process could seriously suffer.

### Related topics

- [Managing nodes on page 2617](#)
- [Geometry request configuration on page 2624](#)

## Scheduling cores

In TORQUE 2.4 and later, you can request specific cores on a node at job submission by using geometry requests. To use this feature, specify the [procs\\_bitmap](#) resource request of `qsub-1` (see [qsub](#)) at job submission.

For details about scheduling cores, see these topics:

- [Geometry request configuration on page 2624](#)
- [Geometry request usage on page 2625](#)
- [Geometry request considerations on page 2625](#)

## Geometry request configuration

A Linux kernel of 2.6 or later is required to use geometry requests, because this feature uses Linux cpusets in its implementation. In order to use this feature, the `cpuset` directory has to be mounted. For more information on how to mount the `cpuset` directory, see [Linux cpuset support on page 2623](#). If the operating environment is suitable for geometry requests, configure TORQUE with the `--enable-geometry-requests` option.

```
> ./configure --prefix=/home/john/torque --enable-geometry-requests
```

TORQUE is configured to install to `/home/john/torque` and to enable the geometry requests feature.

**i** The geometry request feature uses a subset of the cpusets feature. When you configure TORQUE using `--enable-cpuset` and `--enable-geometry-requests` at the same time, and use `-l procs_bitmap=X`, the job will get the requested cpuset. Otherwise, the job is treated as if only `--enable-cpuset` was configured.

#### Related topics

- [Scheduling cores on page 2624](#)

## Geometry request usage

Once enabled, users can submit jobs with a geometry request by using the `procs_bitmap=<string>` resource request. [procs\\_bitmap](#) requires a numerical string made up of 1's and 0's. A 0 in the bitmap means the job cannot run on the core that matches the 0's index in the bitmap. The index is in reverse order of the number of cores available. If a job is submitted with `procs_bitmap=1011`, then the job requests a node with four free cores, and uses only cores one, two, and four.

**i** The geometry request feature requires a node that has all cores free. A job with a geometry request cannot run on a node that has cores that are busy, even if the node has more than enough cores available to run the job.

```
qsub -l procs_bitmap=0011 ossl.sh
```

The job **ossl.sh** is submitted with a geometry request of **0011**.

In the above example, the submitted job can run only on a node that has four cores. When a suitable node is found, the job runs exclusively on cores one and two.

#### Related topics

- [Scheduling cores on page 2624](#)

## Geometry request considerations

As previously stated, jobs with geometry requests require a node with all of its cores available. After the job starts running on the requested cores, the node cannot run other jobs, even if the node has enough free cores to meet the requirements of the other jobs. Once the geometry requesting job is done, the node is available to other jobs again.

#### Related topics

- [Scheduling cores on page 2624](#)

## Scheduling accelerator hardware

TORQUE works with accelerators (such as NVIDIA GPUs and Intel MICs) and can collect and report metrics from them or submit workload to them. This feature requires the use of the Moab scheduler. Refer to the [Accelerators on page 815](#) for information on configuring accelerators in TORQUE.

## Setting server policies

This section explains how to set up and configure your queue. It lists the queue attributes and describes how to set up a routing queue. This section also explains how to set up TORQUE to run in high availability mode. For details, see these topics:

- [Queue configuration on page 2626](#)
- [Server high availability on page 2640](#)

### Queue configuration

Under TORQUE, queue configuration is accomplished using the [Server high availability](#) command. With this tool, the first step is to create the queue. This is accomplished using the `create` subcommand of `qmgr` as in the following example:

```
> qmgr -c "create queue batch queue_type=execution"
```

Once created, the queue must be configured to be operational. At a minimum, this includes setting the options **started** and **enabled**. Further configuration is possible using any combination of the attributes listed in what follows.

For Boolean attributes, *T*, *t*, *1*, *Y*, and *y* are all synonymous with "TRUE," and *F*, *f*, *0*, *N*, and *n* all mean "FALSE."

For [queue\\_type](#), *E* and *R* are synonymous with "Execution" and "Routing" (respectively).

See these topics for more details:

- [Queue attributes on page 2626](#)
- [Example queue configuration on page 2637](#)
- [Setting a default queue on page 2638](#)
- [Mapping a queue to subset of resources on page 2638](#)
- [Creating a routing queue on page 2638](#)

Related topics

- [Appendix B: Server parameters on page 2766](#)
- [qalter on page 2709](#) - command which can move jobs from one queue to another

### Queue attributes

This section lists the following queue attributes:

- [acl\\_groups on page 2627](#)
- [acl\\_group\\_enable on page 2628](#)
- [acl\\_group\\_sloppy on page 2628](#)

- [acl\\_hosts](#) on page 2628
- [acl\\_host\\_enable](#) on page 2629
- [acl\\_logic\\_or](#) on page 2629
- [acl\\_users](#) on page 2629
- [acl\\_user\\_enable](#) on page 2630
- [disallowed\\_types](#) on page 2630
- [enabled](#) on page 2630
- [features\\_required](#) on page 2631
- [keep\\_completed](#) on page 2631
- [kill\\_delay](#) on page 2631
- [max\\_queueable](#) on page 2632
- [max\\_running](#) on page 2632
- [max\\_user\\_queueable](#) on page 2632
- [max\\_user\\_run](#) on page 2633
- [priority](#) on page 2633
- [queue\\_type](#) on page 2633
- [required\\_login\\_property](#) on page 2633
- [resources\\_available](#) on page 2634
- [resources\\_default](#) on page 2634
- [resources\\_max](#) on page 2634
- [resources\\_min](#) on page 2635
- [route\\_destinations](#) on page 2635
- [started](#) on page 2635

This section also lists some queue resource limits (see [Assigning queue resource limits](#) on page 2636).



For Boolean attributes, *T*, *t*, *1*, *Y*, and *y* are all synonymous with "TRUE," and *F*, *f*, *0*, *N*, and *n* all mean "FALSE."

acl_groups	
<b>Format</b>	<GROUP>[@<HOST>][+<USER>[@<HOST>]]...
<b>Default</b>	---

acl_groups	
<b>Description</b>	Specifies the list of groups which may submit jobs to the queue. If <code>acl_group_enable</code> is set to true, only users with a primary group listed in <code>acl_groups</code> may utilize the queue. <p><b>i</b> If the <code>PBSACLUSEGROUPLIST</code> variable is set in the <code>pbs_server</code> environment, <code>acl_groups</code> checks against all groups of which the job user is a member.</p>
<b>Example</b>	<pre>&gt; qmgr -c "set queue batch acl_groups=staff" &gt; qmgr -c "set queue batch acl_groups+=ops@h1" &gt; qmgr -c "set queue batch acl_groups+=staff@h1"</pre> <p><b>i</b> Used in conjunction with <a href="#">acl_group_enable</a>.</p>

acl_group_enable	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If <b>TRUE</b> , constrains TORQUE to only allow jobs submitted from groups specified by the <a href="#">acl_groups</a> parameter.
<b>Example</b>	<pre>qmgr -c "set queue batch acl_group_enable=true"</pre>

acl_group_sloppy	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If <b>TRUE</b> , <a href="#">acl_groups</a> will be checked against all groups of which the job users is a member.
<b>Example</b>	---

acl_hosts	
<b>Format</b>	<HOST>[+<HOST>]...
<b>Default</b>	---

acl_hosts	
<b>Description</b>	Specifies the list of hosts that may submit jobs to the queue.
<b>Example</b>	<pre>qmgr -c "set queue batch acl_hosts=h1+h1+h1"</pre> <p> Used in conjunction with <a href="#">acl_host_enable</a>.</p>

acl_host_enable	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If <b>TRUE</b> , constrains TORQUE to only allow jobs submitted from hosts specified by the <a href="#">acl_hosts</a> parameter.
<b>Example</b>	<pre>qmgr -c "set queue batch acl_host_enable=true"</pre>

acl_logic_or	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	If <b>TRUE</b> , user and group acls are logically OR'd together, meaning that either acl may be met to allow access. If FALSE or unset, then both acls are AND'd, meaning that both acls must be satisfied.
<b>Example</b>	<pre>qmgr -c "set queue batch acl_logic_or=true"</pre>

acl_users	
<b>Format</b>	<USER> [@<HOST>][+<USER>[@<HOST>]]...
<b>Default</b>	---
<b>Description</b>	Specifies the list of users who may submit jobs to the queue. If <a href="#">acl_user_enable</a> is set to <b>TRUE</b> , only users listed in <code>acl_users</code> may use the queue.

## acl\_users

## Example

```
> qmgr -c "set queue batch acl_users=john"
> qmgr -c "set queue batch acl_users+=steve@h1"
> qmgr -c "set queue batch acl_users+=stevek@h1"
```



Used in conjunction with [acl\\_user\\_enable](#).

## acl\_user\_enable

## Format

&lt;BOOLEAN&gt;

## Default

FALSE

## Description

If **TRUE**, constrains TORQUE to only allow jobs submitted from users specified by the [acl\\_users](#) parameter.

## Example

```
qmgr -c "set queue batch acl_user_enable=true"
```

## disallowed\_types

## Format

&lt;type&gt;[+&lt;type&gt;]...

## Default

---

## Description

Specifies classes of jobs that are not allowed to be submitted to this queue. Valid types are interactive, batch, rerunable, nonrerunable, fault\_tolerant (as of version 2.4.0 and later), fault\_intolerant (as of version 2.4.0 and later), and job\_array (as of version 2.4.1 and later).

## Example

```
qmgr -c "set queue batch disallowed_types = interactive"
qmgr -c "set queue batch disallowed_types += job_array"
```

## enabled

## Format

&lt;BOOLEAN&gt;

## Default

FALSE

## Description

Specifies whether the queue accepts new job submissions.

## enabled

### Example

```
qmgr -c "set queue batch enabled=true"
```

## features\_required

### Format

feature1[feature2[,feature3...]]

### Default

---

### Description

Specifies that all jobs in this queue will require these features in addition to any they may have requested. A feature is a synonym for a property.

### Example

```
qmgr -c 's q batch features_required=fast'
```

## keep\_completed

### Format

<INTEGER>

### Default

0

### Description

Specifies the number of seconds jobs should be held in the Completed state after exiting. For more information, see [Keeping completed jobs on page 2600](#).

### Example

```
qmgr -c "set queue batch keep_completed=120"
```

## kill\_delay

### Format

<INTEGER>

### Default

2

### Description

Specifies the number of seconds between sending a SIGTERM and a SIGKILL to a job in a specific queue that you want to cancel. It is possible that the job script, and any child processes it spawns, can receive several SIGTERM signals before the SIGKILL signal is received.



All MOMs must be configured with `$exec with exec true` in order for `kill_delay` to work, even when relying on default `kill_delay` settings.

**kill\_delay****Example**

```
qmgr -c "set queue batch kill_delay=30"
```

**max\_queuable****Format**

&lt;INTEGER&gt;

**Default**

unlimited

**Description**

Specifies the maximum number of jobs allowed in the queue at any given time (includes idle, running, and blocked jobs).

**Example**

```
qmgr -c "set queue batch max_queuable=20"
```

**max\_running****Format**

&lt;INTEGER&gt;

**Default**

unlimited

**Description**

Specifies the maximum number of jobs in the queue allowed to run at any given time.

**Example**

```
qmgr -c "set queue batch max_running=20"
```

**max\_user\_queuable****Format**

&lt;INTEGER&gt;

**Default**

unlimited

**Description**

Specifies the maximum number of jobs, per user, allowed in the queue at any given time (includes idle, running, and blocked jobs). Version 2.1.3 and greater.

**Example**

```
qmgr -c "set queue batch max_user_queuable=20"
```

max_user_run	
<b>Format</b>	<INTEGER>
<b>Default</b>	unlimited
<b>Description</b>	Specifies the maximum number of jobs, per user, in the queue allowed to run at any given time.
<b>Example</b>	<pre>qmgr -c "set queue batch max_user_run=10"</pre>

priority	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the priority value associated with the queue.
<b>Example</b>	<pre>qmgr -c "set queue batch priority=20"</pre>

queue_type	
<b>Format</b>	One of <i>e</i> , <i>execution</i> , <i>r</i> , or <i>route</i> (see <a href="#">Creating a routing queue on page 2638</a> )
<b>Default</b>	---
<b>Description</b>	Specifies the queue type. <div style="border: 1px solid red; background-color: #ffe6e6; padding: 5px; margin-top: 10px;">  This value must be explicitly set for all queues. </div>
<b>Example</b>	<pre>qmgr -c "set queue batch queue_type=execution"</pre>

required_login_property	
<b>Format</b>	<STRING>
<b>Default</b>	---

## required\_login\_property

**Description** Adds the specified login property as a requirement for all jobs in this queue.

**Example** `qmgr -c 's q <queuename> required_login_property=INDUSTRIAL'`

## resources\_available

**Format** <STRING>

**Default** ---

**Description** Specifies cumulative resources available to all jobs running in the queue. See [qsub will not allow the submission of jobs requesting many processors on page 2675](#) for more information.

**Example** `qmgr -c "set queue batch resources_available.nodect=20"`



You must restart pbs\_server for changes to take effect.

Also, resources\_available is constrained by the smallest of queue.resources\_available and server.resources\_available.

## resources\_default

**Format** <STRING>

**Default** ---

**Description** Specifies default resource requirements for jobs submitted to the queue.

**Example** `qmgr -c "set queue batch resources_default.walltime=3600"`

## resources\_max

**Format** <STRING>

**Default** ---

resources_max	
<b>Description</b>	Specifies the maximum resource limits for jobs submitted to the queue.
<b>Example</b>	<pre>qmgr -c "set queue batch resources_max.nodect=16"</pre>

resources_min	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies the minimum resource limits for jobs submitted to the queue.
<b>Example</b>	<pre>qmgr -c "set queue batch resources_min.nodect=2"</pre>

route_destinations	
<b>Format</b>	<queue>[@<host>]
<b>Default</b>	---
<b>Description</b>	Specifies the potential destination queues for jobs submitted to the associated routing queue. <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;">  This attribute is only valid for routing queues (see <a href="#">Creating a routing queue on page 2638</a>). </div>
<b>Example</b>	<pre>&gt; qmgr -c "set queue route route_destinations=fast" &gt; qmgr -c "set queue route route_destinations+=slow" &gt; qmgr -c "set queue route route_destinations+=medium@hostname"</pre> <p>To set multiple queue specifications, use multiple commands:</p> <pre>&gt; qmgr -c 's s route_destinations=batch' &gt; qmgr -c 's s route_destinations+=long' &gt; qmgr -c 's s route_destinations+=short'</pre>

started	
<b>Format</b>	<BOOLEAN>

started	
<b>Default</b>	FALSE
<b>Description</b>	Specifies whether jobs in the queue are allowed to execute.
<b>Example</b>	<pre>qmgr -c "set queue batch started=true"</pre>

## Assigning queue resource limits

Administrators can use resources limits to help direct what kind of jobs go to different queues. There are four queue attributes where resource limits can be set: [resources\\_available](#), [resources\\_default](#), [resources\\_max](#), and [resources\\_min](#). The list of supported resources that can be limited with these attributes are *arch*, *mem*, *nodect*, *nodes*, *procct*, *pvmem*, *vmem*, and *walltime*.

Resource	Format	Description
<b>arch</b>	string	Specifies the administrator defined system architecture required.
<b>mem</b>	<a href="#">size</a>	Amount of physical memory used by the job. (Ignored on Darwin, Digital Unix, Free BSD, HPUX 11, IRIX, NetBSD, and SunOS. Also ignored on Linux if number of nodes is not 1. Not implemented on AIX and HPUX 10.)
<b>ncpus</b>	integer	Sets the number of processors in one task where a task cannot span nodes. <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; background-color: #E6F2FF; margin-top: 10px;"><b>i</b> You cannot request both <code>ncpus</code> and <code>nodes</code> in the same queue.</div>
<b>nodect</b>	integer	Sets the number of nodes available. By default, TORQUE will set the number of nodes available to the number of nodes listed in the <code>\$TORQUE_HOME/server_priv/nodes</code> file. <code>nodect</code> can be set to be greater than or less than that number. Generally, it is used to set the node count higher than the number of physical nodes in the cluster.
<b>nodes</b>	integer	Specifies the number of nodes.
<b>procct</b>	integer	Sets limits on the total number of execution slots (procs) allocated to a job. The number of procs is calculated by summing the products of all node and ppn entries for a job. For example <code>qsub -l nodes=2:ppn=2+3:ppn=4 job.sh</code> would yield a <code>procct</code> of 16. $2*2$ (2:ppn=2) + $3*4$ (3:ppn=4).
<b>pvmem</b>	<a href="#">size</a>	Amount of virtual memory used by any single process in a job.

Resource	Format	Description
<b>vmem</b>	<a href="#">size</a>	Amount of virtual memory used by all concurrent processes in the job.
<b>walltime</b>	seconds, or [[HH:] MM:]SS	Amount of real time during which a job can be in a running state.

### *size*

The size format specifies the maximum amount in terms of bytes or words. It is expressed in the form *integer[suffix]*. The suffix is a multiplier defined in the following table ("b" means bytes [the default] and "w" means words). The size of a word is calculated on the execution server as its word size.

Suffix		Multiplier
b	w	1
kb	kw	1024
mb	mw	1,048,576
gb	gw	1,073,741,824
tb	tw	1,099,511,627,776

### Related topics

- [Queue configuration on page 2626](#)
- [Example queue configuration on page 2637](#)

## Example queue configuration

The following series of **qmgr** commands will create and configure a queue named batch:

```
qmgr -c "create queue batch queue_type=execution"
qmgr -c "set queue batch started=true"
qmgr -c "set queue batch enabled=true"
qmgr -c "set queue batch resources_default.nodes=1"
qmgr -c "set queue batch resources_default.walltime=3600"
```

This queue will accept new jobs and, if not explicitly specified in the job, will assign a nodecount of 1 and a walltime of 1 hour to each job.

### Related topics

- [Queue configuration on page 2626](#)

## Setting a default queue

By default, a job must explicitly specify which queue it is to run in. To change this behavior, the server parameter [default\\_queue](#) may be specified as in the following example:

```
qmgr -c "set server default_queue=batch"
```

### Related topics

- [Queue configuration on page 2626](#)

## Mapping a queue to subset of resources

TORQUE does not currently provide a simple mechanism for mapping queues to nodes. However, schedulers such as [Moab](#) and [Maui](#) can provide this functionality.

The simplest method is using `default_resources.neednodes` on an execution queue, setting it to a particular node attribute. Maui/Moab will use this information to ensure that jobs in that queue will be assigned nodes with that attribute. For example, suppose we have some nodes bought with money from the chemistry department, and some nodes paid by the biology department.

```
$TORQUE_HOME/server_priv/nodes:
node01 np=2 chem
node02 np=2 chem
node03 np=2 bio
node04 np=2 bio
qmgr:
set queue chem resources_default.neednodes=chem
set queue bio resources_default.neednodes=bio
```

**i** This example does not preclude other queues from accessing those nodes. One solution is to use some other generic attribute with all other nodes and queues.

More advanced configurations can be made with standing reservations and QoSs.

### Related topics

- [Queue configuration on page 2626](#)

## Creating a routing queue

A routing queue will steer a job to a destination queue based on job attributes and queue constraints. It is set up by creating a queue of [queue\\_type](#) "Route" with a [route\\_destinations](#) attribute set, as in the following example.

```

qmgr

# routing queue
create queue route
set queue route queue_type = Route
set queue route route_destinations = reg_64
set queue route route_destinations += reg_32
set queue route route_destinations += reg_
set queue route enabled = True
set queue route started = True

# queue for jobs using 1-15 nodes
create queue reg
set queue reg queue_type = Execution
set queue reg resources_min.ncpus = 1
set queue reg resources_min.nodect = 1
set queue reg resources_default.ncpus = 1
set queue reg resources_default.nodes = 1
set queue reg enabled = True
set queue reg started = True

# queue for jobs using 16-31 nodes
create queue reg_32
set queue reg_32 queue_type = Execution
set queue reg_32 resources_min.ncpus = 31
set queue reg_32 resources_min.nodes = 16
set queue reg_32 resources_default.walltime = 12:00:00
set queue reg_32 enabled = True
set queue reg_32 started = True

# queue for jobs using 32+ nodes
create queue reg_64
set queue reg_64 queue_type = Execution
set queue reg_64 resources_min.ncpus = 63
set queue reg_64 resources_min.nodes = 32
set queue reg_64 resources_default.walltime = 06:00:00
set queue reg_64 enabled = True
set queue reg_64 started = True

# have all jobs go through the routing queue
set server default_queue = batch
set server resources_default.ncpus = 1
set server resources_default.walltime = 24:00:00
...

```

In this example, the compute nodes are dual processors and default walltimes are set according to the number of processors/nodes of a job. Jobs with 32 nodes (63 processors) or more will be given a default walltime of 6 hours. Also, jobs with 16-31 nodes (31-62 processors) will be given a default walltime of 12 hours. All other jobs will have the server default walltime of 24 hours.

The ordering of the `route_destinations` is important. In a routing queue, a job is assigned to the first possible destination queue based on the [resources\\_max](#), [resources\\_min](#), [acl\\_users](#), and [acl\\_groups](#) attributes. In the preceding example, the attributes of a single processor job would first be checked against the `reg_64` queue, then the `reg_32` queue, and finally the `reg` queue.

Adding the following settings to the earlier configuration elucidates the queue resource requirements:

```

qmgr

set queue reg resources_max.ncpus = 30
set queue reg resources_max.nodect = 15
set queue reg_16 resources_max.ncpus = 62
set queue reg_16 resources_max.nodect = 31

```

The time of enforcement of server and queue defaults is important in this example. TORQUE applies server and queue defaults differently in job centric and queue centric modes. For job centric mode, TORQUE waits to apply the server and queue defaults until the job is assigned to its final execution queue. For queue centric mode, it enforces server defaults before it is placed in the routing queue. In either mode, queue defaults override the server defaults. TORQUE defaults to job centric mode. To set queue centric mode, set `queue_centric_limits`, as in what follows:

```
qmgr
set server queue_centric_limits = true
```

An artifact of job centric mode is that if a job does not have an attribute set, the server and routing queue defaults are not applied when queue resource limits are checked. Consequently, a job that requests 32 nodes (not `ncpus=32`) will not be checked against a `min_resource.ncpus` limit. Also, for the preceding example, a job without any attributes set will be placed in the `reg_64` queue, since the server `ncpus` default will be applied after the job is assigned to an execution queue.

**i** Routine queue defaults are not applied to job attributes in versions 2.1.0 and before.

**i** If the error message "qsub: Job rejected by all possible destinations" is reported when submitting a job, it may be necessary to add queue location information, (i.e., in the routing queue's `route_destinations` attribute, change "batch" to "batch@localhost").

#### Related topics

- [Queue configuration on page 2626](#)
- [Queue attributes on page 2626](#)

## Server high availability

You can now run TORQUE in a redundant or high availability mode. This means that there can be multiple instances of the server running and waiting to take over processing in the event that the currently running server fails.

**i** The high availability feature is available in the 2.3 and later versions of TORQUE. TORQUE 2.4 includes several enhancements to high availability (see [Server high availability on page 2640](#)).

**i** Contact Adaptive Computing before attempting to implement any type of high availability.

For more details, see these sections:

- [Redundant server host machines on page 2641](#)
- [Server high availability on page 2640](#)
- [Enhanced high availability with Moab on page 2642](#)
- [How commands select the correct server host on page 2642](#)

- [Job names on page 2643](#)
- [Persistence of the pbs\\_server process on page 2643](#)
- [High availability of the NFS server on page 2643](#)
- [Installing TORQUE in high availability mode on page 2643](#)
- [Installing TORQUE in high availability mode on headless nodes on page 2648](#)
- [Example setup of high availability on page 2652](#)

## Redundant server host machines

High availability enables TORQUE to continue running even if pbs\_server is brought down. This is done by running multiple copies of pbs\_server which have their torque/server\_priv directory mounted on a shared file system. The torque/server\_name must include the host names of all nodes that run pbs\_server. All MOM nodes also must include the host names of all nodes running pbs\_server in their torque/server\_name file. The syntax of the torque/server\_name is a comma delimited list of host names.

For example:

```
host1,host2,host3
```



When configuring high availability, do not use \$pbsserver to specify the host names. You must use the \$TORQUEHOMEDIR/server\_name file.

All instances of pbs\_server need to be started with the --ha command line option that allows the servers to run at the same time. Only the first server to start will complete the full startup. The second server to start will block very early in the startup when it tries to lock the file torque/server\_priv/server.lock. When the second server cannot obtain the lock, it will spin in a loop and wait for the lock to clear. The sleep time between checks of the lock file is one second.

Notice that not only can the servers run on independent server hardware, there can also be multiple instances of the pbs\_server running on the same machine. This was not possible before as the second one to start would always write an error and quit when it could not obtain the lock.

## Enabling high availability

To use high availability, you must start each instance of pbs\_server with the --ha option.

Prior to version 4.0, TORQUE with HA was configured with an --enable-high-availability option. That option is no longer required.

Three server options help manage high availability. The server parameters are [lock\\_file](#), [lock\\_file\\_update\\_time](#), and [lock\\_file\\_check\\_time](#).

The lock\_file option allows the administrator to change the location of the lock file. The default location is torque/server\_priv. If the lock\_file option is used, the new location must be on the shared partition so all servers have access.

The `lock_file_update_time` and `lock_file_check_time` parameters are used by the servers to determine if the primary server is active. The primary `pbs_server` will update the lock file based on the `lock_file_update_time` (default value of 3 seconds). All backup `pbs_servers` will check the lock file as indicated by the `lock_file_check_time` parameter (default value of 9 seconds). The `lock_file_update_time` must be less than the `lock_file_check_time`. When a failure occurs, the backup `pbs_server` takes up to the `lock_file_check_time` value to take over.

```
> qmgr -c "set server lock_file_check_time=5"
```

In the above example, after the primary `pbs_server` goes down, the backup `pbs_server` takes up to 5 seconds to take over. It takes additional time for all MOMs to switch over to the new `pbs_server`.

**i** The clock on the primary and redundant servers must be synchronized in order for high availability to work. Use a utility such as NTP to ensure your servers have a synchronized time.

**i** Do not use anything but a plain simple NFS fileshare that is not used by anybody or anything else (i.e., only Moab can use the fileshare).

**i** Do not use any general-purpose NAS, do not use any parallel file system, and do not use company-wide shared infrastructure to set up Moab high availability using "native" high availability.

## Enhanced high availability with Moab

When TORQUE is run with an external scheduler such as Moab, and the `pbs_server` is not running on the same host as Moab, `pbs_server` needs to know where to find the scheduler. To do this, use the `-l` option as demonstrated in the example below (the port is required and the default is 15004).

```
> pbs_server -l <moabhost:port>
```

If Moab is running in HA mode, add a `-l` option for each redundant server.

```
> pbs_server -l <moabhost1:port> -l <moabhost2:port>
```

If `pbs_server` and Moab run on the same host, use the `--ha` option as demonstrated in the example below.

```
> pbs_server --ha
```

The root user of each Moab host must be added to the [operators](#) and [managers](#) lists of the server. This enables Moab to execute root level operations in TORQUE.

## How commands select the correct server host

The various commands that send messages to `pbs_server` usually have an option of specifying the server name on the command line, or if none is specified will use the default server name. The default server name comes either from the environment variable `PBS_DEFAULT` or from the file `torque/server_name`.

When a command is executed and no explicit server is mentioned, an attempt is made to connect to the first server name in the list of hosts from `PBS_DEFAULT` or `torque/server_name`. If this fails, the next server name is tried. If all servers in the list are unreachable, an error is returned and the command fails.

Note that there is a period of time after the failure of the current server during which the new server is starting up where it is unable to process commands. The new server must read the existing configuration and job information from the disk, so the length of time that commands cannot be received varies. Commands issued during this period of time might fail due to timeouts expiring.

## Job names

Job names normally contain the name of the host machine where `pbs_server` is running. When job names are constructed, only the server name in `$PBS_DEFAULT` or the first name from the server specification list, `$TORQUE_HOME/server_name`, is used in building the job name.

## Persistence of the `pbs_server` process

The system administrator must ensure that `pbs_server` continues to run on the server nodes. This could be as simple as a *cron* job that counts the number of `pbs_server`'s in the process table and starts some more if needed.

## High availability of the NFS server



Before installing a specific NFS HA solution please contact Adaptive Computing Support for a detailed discussion on NFS HA type and implementation path.

One consideration of this implementation is that it depends on NFS file system also being redundant. NFS can be set up as a redundant service. See the following.

- [Setting Up A Highly Available NFS Server](#)
- [Making NFS Work On Your Network](#)
- [Sourceforge Linux NFS FAQ](#)
- [NFS v4 main site](#)

There are also other ways to set up a shared file system. See the following:

- [Red Hat Global File System](#)
- [Data sharing with a GFS storage cluster](#)

## Installing TORQUE in high availability mode

The following procedure demonstrates a TORQUE installation in high availability (HA) mode.

## Requirements

- gcc (GCC) 4.1.2
- BASH shell
- Servers configured the following way:
  - 2 main servers with identical architecture:
    - server1 — Primary server running TORQUE with a shared file system (this example uses NFS)
    - server2 — Secondary server running with TORQUE with a shared file system (this example uses NFS)
  - fileServer — Shared file system (this example uses NFS)
  - Compute nodes

## To install TORQUE in HA mode

1. Stop all firewalls or update your firewall to allow traffic from TORQUE services.

```
> service iptables stop
> chkconfig iptables off
```

If you are unable to stop the firewall due to infrastructure restriction, open the following ports:

- 15001 [tcp,udp]
- 15002 [tcp,udp]
- 15003 [tcp,udp]

2. Disable SELinux

```
> vi /etc/sysconfig/selinux
SELINUX=disabled
```

3. Update your main ~/.bashrc profile to ensure you are always referencing the applications to be installed on all servers.

```
# TORQUE
export TORQUEHOME=/var/spool/torque

# Library Path
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:${TORQUEHOME}/lib

# Update system paths
export PATH=${TORQUEHOME}/bin:${TORQUEHOME}/sbin:${PATH}
```

4. Verify server1 and server2 are resolvable via either DNS or looking for an entry in the /etc/hosts file.
5. Configure the NFS Mounts by following these steps:

- a. Create mount point folders on fileServer.

```
fileServer# mkdir -m 0755 /var/spool/torque
fileServer# mkdir -m 0750 /var/spool/torque/server_priv
```

- b. Update /etc/exports on fileServer. The IP addresses should be that of server2.

```
/var/spool/torque/server_priv 192.168.0.0/255.255.255.0(rw, sync, no_root_squash)
```

- c. Update the list of NFS exported file systems.

```
fileServer# exportfs -r
```

6. If the NFS daemons are not already running on fileServer, start them.

```
> systemctl restart rpcbind.service
> systemctl start nfs-server.service
> systemctl start nfs-lock.service
> systemctl start nfs-idmap.service
```

7. Mount the exported file systems on server1 by following these steps:

- a. Create the directory reference and mount them.

```
server1# mkdir /var/spool/torque/server_priv
```

Repeat this process for server2.

- b. Update /etc/fstab on server1 to ensure that NFS mount is performed on startup.

```
fileServer:/var/spool/torque/server_priv /var/spool/torque/server_priv nfs
rsize= 8192, wsize=8192, timeo=14, intr
```

Repeat this step for server2.

8. Install TORQUE by following these steps:

- a. Download and extract TORQUE 5.0.0 on server1.

```
server1# wget http://github.com/adaptivecomputing/torque/ branches/5.0.0/torque-
5.0.0.tar.gz
server1# tar -xvzf torque-5.0.0.tar.gz
```

- b. Navigate to the TORQUE directory and compile TORQUE on server1.

```
server1# configure
server1# make
server1# make install
server1# make packages
```

- c. If the installation directory is shared on both head nodes, then run `make install` on server1.

```
server1# make install
```

If the installation directory is not shared, repeat step 8a-b (downloading and installing TORQUE) on server2.

9. Start `trqauthd`.

```
server1# /etc/init.d/trqauthd start
```

## 10. Configure TORQUE for HA.

- a. List the host names of all nodes that run `pbs_server` in the `torque/server_name` file. You must also include the host names of all nodes running `pbs_server` in the `torque/server_name` file of each MOM node. The syntax of `torque/server_name` is a comma-delimited list of host names.

```
server1
server2
```

- b. Create a simple queue configuration for TORQUE job queues on server1.

```
server1# pbs_server -t create
server1# qmgr -c "set server scheduling=true"
server1# qmgr -c "create queue batch queue_type=execution"
server1# qmgr -c "set queue batch started=true"
server1# qmgr -c "set queue batch enabled=true"
server1# qmgr -c "set queue batch resources_default.nodes=1"
server1# qmgr -c "set queue batch resources_default.walltime=3600"
server1# qmgr -c "set server default_queue=batch"
```

**i** Because `server_priv/*` is a shared drive, you do not need to repeat this step on `server2`.

- c. Add the root users of TORQUE to the TORQUE configuration as an operator and manager.

```
server1# qmgr -c "set server managers += root@server1"
server1# qmgr -c "set server managers += root@server2"
server1# qmgr -c "set server operators += root@server1"
server1# qmgr -c "set server operators += root@server2"
```

**i** Because `server_priv/*` is a shared drive, you do not need to repeat this step on `Server 2`.

- d. You must update the lock file mechanism for TORQUE in order to determine which server is the primary. To do so, use the `lock_file_update_time` and `lock_file_check_time` parameters. The primary `pbs_server` will update the lock file based on the specified `lock_file_update_time` (default value of 3 seconds). All backup `pbs_servers` will check the lock file as indicated by the `lock_file_check_time` parameter (default value of 9 seconds). The `lock_file_update_time` must be less than the `lock_file_check_time`. When a failure occurs, the backup `pbs_server` takes up to the `lock_file_check_time` value to take over.

```
server1# qmgr -c "set server lock_file_check_time=5"
server1# qmgr -c "set server lock_file_update_time=3"
```

**i** Because `server_priv/*` is a shared drive, you do not need to repeat this step on `server2`.

- e. List the servers running `pbs_server` in the TORQUE `acl_hosts` file.

```
server1# qmgr -c "set server acl_hosts += server1"
server1# qmgr -c "set server acl_hosts += server2"
```

**i** Because `server_priv/*` is a shared drive, you do not need to repeat this step on `server2`.

- f. Restart the running `pbs_server` in HA mode.

```
server1# qterm
```

- g. Start the `pbs_server` on the secondary server.

```
server1# pbs_server --ha -l server2:port
server2# pbs_server --ha -l server1:port
```

11. Check the status of TORQUE in HA mode.

```
server1# qmgr -c "p s"
server2# qmgr -c "p s"
```

*The commands above returns all settings from the active TORQUE server from either node.*

Drop one of the `pbs_servers` to verify that the secondary server picks up the request.

```
server1# qterm
server2# qmgr -c "p s"
```

Stop the `pbs_server` on `server2` and restart `pbs_server` on `server1` to verify that both nodes can handle a request from the other.

12. Install a `pbs_mom` on the compute nodes.

- a. Copy the install scripts to the compute nodes and install.
- b. Navigate to the shared source directory of TORQUE and run the following:

```
node1# torque-package-mom-linux-x86_64.sh --install
node2# torque-package-clients-linux-x86_64.sh --install
```

Repeat this for each compute node. Verify that the `/var/pool/ torque/server-name` file shows all your compute nodes.

- c. On `server1` or `server2`, configure the `nodes` file to identify all available MOMs. To do so, edit the `/var/spool/torque/server_priv/nodes` file.

```
node1 np=2
node2 np=2
```

**i** Change the `np` flag to reflect number of available processors on that node.

- d. Recycle the `pbs_servers` to verify that they pick up the MOM configuration.

```
server1# qterm; pbs_server --ha -l server2:port
server2# qterm; pbs_server --ha -l server1:port
```

- e. Start the `pbs_mom` on each execution node.

```
node5# pbs_mom
node6# pbs_mom
```

## Installing TORQUE in high availability mode on headless nodes

The following procedure demonstrates a TORQUE installation in high availability (HA) mode on nodes with no local hard drive.

### Requirements

- gcc (GCC) 4.1.2
- BASH shell
- Servers (these cannot be two VMs on the same hypervisor) configured the following way:
  - 2 main servers with identical architecture
    - `server1` — Primary server running TORQUE with a file system share (this example uses NFS)
    - `server2` — Secondary server running with TORQUE with a file system share (this example uses NFS)
  - Compute nodes
  - `fileServer` — A shared file system server (this example uses NFS)

### To install TORQUE in HA mode on a node with no local hard drive

1. Stop all firewalls or update your firewall to allow traffic from TORQUE services.

```
> service iptables stop
> chkconfig iptables off
```

If you are unable to stop the firewall due to infrastructure restriction, open the following ports:

- 15001 [tcp,udp]
- 15002 [tcp,udp]
- 15003 [tcp,udp]

2. Disable SELinux

```
> vi /etc/sysconfig/selinux
SELINUX=disabled
```

3. Update your main `~/ .bashrc` profile to ensure you are always referencing the applications to be installed on all servers.

```
# TORQUE
export TORQUEHOME=/var/spool/torque

# Library Path

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:${TORQUEHOME}/lib

# Update system paths
export PATH=${TORQUEHOME}/bin:${TORQUEHOME}/sbin:${PATH}
```

4. Verify `server1` and `server2` are resolvable via either DNS or looking for an entry in the `/etc/hosts` file.

5. Configure the NFS Mounts by following these steps:

a. Create mount point folders on `fileServer`.

```
fileServer# mkdir -m 0755 /var/spool/torque
```

b. Update `/etc/exports` on `fileServer`. The IP addresses should be that of `server2`.

```
/var/spool/torque/ 192.168.0.0/255.255.255.0(rw,sync,no_root_squash)
```

c. Update the list of NFS exported file systems.

```
fileServer# exportfs -r
```

6. If the NFS daemons are not already running on `fileServer`, start them.

```
> systemctl restart rpcbind.service
> systemctl start nfs-server.service
> systemctl start nfs-lock.service
> systemctl start nfs-idmap.service
```

7. Mount the exported file systems on `server1` by following these steps:

a. Create the directory reference and mount them.

```
server1# mkdir /var/spool/torque
```

Repeat this process for `server2`.

b. Update `/etc/fstab` on `server1` to ensure that NFS mount is performed on startup.

```
fileServer:/var/spool/torque/server_priv /var/spool/torque/server_priv nfs
rsize= 8192, wsize=8192, timeo=14, intr
```

Repeat this step for `server2`.

8. Install TORQUE by following these steps:

a. Download and extract TORQUE 5.0.0 on `server1`.

```
server1# wget http://github.com/adaptivecomputing/torque/branches/5.0.0/torque-5.0.0.tar.gz
server1# tar -xvzf torque-5.0.0.tar.gz
```

b. Navigate to the TORQUE directory and compile TORQUE with the HA flag on `server1`.

```
server1# configure --prefix=/var/spool/torque
server1# make
server1# make install
server1# make packages
```

- c. If the installation directory is shared on both head nodes, then run `make install` on `server1`.

```
server1# make install
```

If the installation directory is not shared, repeat step 8a-b (downloading and installing TORQUE) on `server2`.

9. Start `trqauthd`.

```
server1# /etc/init.d/trqauthd start
```

10. Configure TORQUE for HA.

- a. List the host names of all nodes that run `pbs_server` in the `torque/server_name` file. You must also include the host names of all nodes running `pbs_server` in the `torque/server_name` file of each MOM node. The syntax of `torque/server_name` is a comma-delimited list of host names.

```
server1,server2
```

- b. Create a simple queue configuration for TORQUE job queues on `server1`.

```
server1# pbs_server -t create
server1# qmgr -c "set server scheduling=true"
server1# qmgr -c "create queue batch queue_type=execution"
server1# qmgr -c "set queue batch started=true"
server1# qmgr -c "set queue batch enabled=true"
server1# qmgr -c "set queue batch resources_default.nodes=1"
server1# qmgr -c "set queue batch resources_default.walltime=3600"
server1# qmgr -c "set server default_queue=batch"
```

**i** Because `TORQUEHOME` is a shared drive, you do not need to repeat this step on `server2`.

- c. Add the root users of TORQUE to the TORQUE configuration as an operator and manager.

```
server1# qmgr -c "set server managers += root@server1"
server1# qmgr -c "set server managers += root@server2"
server1# qmgr -c "set server operators += root@server1"
server1# qmgr -c "set server operators += root@server2"
```

**i** Because `TORQUEHOME` is a shared drive, you do not need to repeat this step on `server2`.

- d. You must update the lock file mechanism for TORQUE in order to determine which server is the primary. To do so, use the `lock_file_update_time` and `lock_file_check_time` parameters. The primary `pbs_server` will update the lock file based on the specified `lock_file_update_time` (default value of 3 seconds). All backup `pbs_servers` will check the lock file as

indicated by the `lock_file_check_time` parameter (default value of 9 seconds). The `lock_file_update_time` must be less than the `lock_file_check_time`. When a failure occurs, the backup `pbs_server` takes up to the `lock_file_check_time` value to take over.

```
server1# qmgr -c "set server lock_file_check_time=5"
server1# qmgr -c "set server lock_file_update_time=3"
```

**i** Because `TORQUEHOME` is a shared drive, you do not need to repeat this step on `server2`.

- e. List the servers running `pbs_server` in the TORQUE `acl_hosts` file.

```
server1# qmgr -c "set server acl_hosts += server1"
server1# qmgr -c "set server acl_hosts += server2"
```

**i** Because `TORQUEHOME` is a shared drive, you do not need to repeat this step on `server2`.

- f. Restart the running `pbs_server` in HA mode.

```
server1# qterm
```

- g. Start the `pbs_server` on the secondary server.

```
server1# pbs_server --ha -l server2:port
server2# pbs_server --ha -l server1:port
```

11. Check the status of TORQUE in HA mode.

```
server1# qmgr -c "p s"
server2# qmgr -c "p s"
```

*The commands above returns all settings from the active TORQUE server from either node.*

Drop one of the `pbs_servers` to verify that the secondary server picks up the request.

```
server1# qterm
server2# qmgr -c "p s"
```

Stop the `pbs_server` on `server2` and restart `pbs_server` on `server1` to verify that both nodes can handle a request from the other.

12. Install a `pbs_mom` on the compute nodes.

- a. On `server1` or `server2`, configure the nodes file to identify all available MOMs. To do so, edit the `/var/spool/torque/server_priv/nodes` file.

```
node1 np=2
node2 np=2
```

**i** Change the `np` flag to reflect number of available processors on that node.

- b. Recycle the `pbs_servers` to verify that they pick up the MOM configuration.

```
server1# qterm; pbs_server --ha -l server2:port
server2# qterm; pbs_server --ha -l server1:port
```

- c. Start the pbs\_mom on each execution node.

```
server1# pbs_mom -d <mom-server1>
server2# pbs_mom -d <mom-server2>
```

## Example setup of high availability

1. The machines running pbs\_server must have access to a shared server\_priv/ directory (usually an NFS share on a MoM).
2. All MoMs must have the same content in their server\_name file. This can be done manually or via an NFS share. The server\_name file contains a comma-delimited list of the hosts that run pbs\_server.

```
# List of all servers running pbs_server
server1,server2
```

3. The machines running pbs\_server must be listed in [acl\\_hosts](#).

```
> qmgr -c "set server acl_hosts += server1"
> qmgr -c "set server acl_hosts += server2"
```

4. Start pbs\_server with the --ha option.

```
[root@server1]$ pbs_server --ha
[root@server2]$ pbs_server --ha
```

### Related topics

- [Setting server policies on page 2626](#)
- [Queue configuration on page 2626](#)

## Setting min\_threads and max\_threads

There are two threadpools in TORQUE, one for background tasks and one for incoming requests from the MOMs and through the API (client commands, Moab, and so forth). The [min\\_threads on page 2778](#) and [max\\_threads on page 2777](#) parameters control the number of total threads used for both, not for each individually. The incoming requests' threadpool has three-quarters of [min\\_threads](#) for its minimum, and three-quarters of [max\\_threads](#) for its maximum, with the background pool receiving the other one-quarter.

Additionally, pbs\_server no longer allows incoming requests to pile up indefinitely. When the threadpool is too busy for incoming requests, it indicates such, returning PBSE\_SERVER\_BUSY with the accompanying message that "Pbs Server is currently too busy to service this request. Please retry this request." The threshold for this message, if the request is from a manager, is that at least two threads be available in the threadpool. If the request comes from a non-manager, 5% of the threadpool must be available for the request to be serviced. Note that availability is calculated based on the maximum threads and not based on the current number of threads allocated.

If an undesirably large number of requests are given a busy response, one option is to increase the number of maximum threads for the threadpool. If the load on the server is already very high, then this is probably not going to help, but if the CPU load is lower, then it may help. Remember that by default the threadpool shrinks down once the extra threads are no longer needed. This is controlled via the [thread\\_idle\\_seconds](#) on [page 2783](#) server parameter.

**i** Any change in the [min\\_threads](#), [max\\_threads](#), or [thread\\_idle\\_seconds](#) parameters requires a restart of `pbs_server` to take effect.

## Integrating schedulers for TORQUE

Selecting the cluster scheduler is an important decision and significantly affects cluster utilization, responsiveness, availability, and intelligence. The default TORQUE scheduler, `pbs_sched`, is very basic and will provide poor utilization of your cluster's resources. Other options, such as Maui Scheduler or Moab Workload Manager, are highly recommended. If you are using [Maui](#) or [Moab](#), see [Moab-TORQUE Integration Guide](#) on [page 1228](#). If using `pbs_sched`, simply start the `pbs_sched` daemon.

**i** If you are installing Moab Cluster Suite, TORQUE and Moab were configured at installation for interoperability and no further action is required.

## Configuring data management

This section contains information about SCP-based data management with TORQUE. It describes how to use TORQUE with NFS and other networked filesystems. It also outlines file staging requirements. For details, see these topics:

- [SCP setup](#) on [page 2653](#)
- [NFS and other networked filesystems](#) on [page 2656](#)
- [File stage-in/stage-out](#) on [page 2657](#)

### SCP setup

To use SCP-based data management, TORQUE must be authorized to migrate data to any of the compute nodes. If this is not already enabled within the cluster, this can be achieved with the process described below. This process enables uni-directional access for a particular user from a *source* host to a *destination* host.

**i** These directions were written using [OpenSSH version 3.6](#) and may not transfer correctly to older versions.

To set up TORQUE for SCP, follow the directions in each of these topics:

- [Generating SSH key on source host on page 2654](#)
- [Copying public SSH key to each destination host on page 2654](#)
- [Configuring the SSH daemon on each destination host on page 2655](#)
- [Validating correct SSH configuration on page 2655](#)
- [Enabling bi-directional SCP access on page 2655](#)
- [Compiling TORQUE to support SCP on page 2656](#)
- [Troubleshooting on page 2656](#)

#### Related topics

- [Configuring data management on page 2653](#)

## Generating SSH key on source host

On the source host as the transfer user, execute the following:

```
> ssh-keygen -t rsa
```

This will prompt for a passphrase (optional) and create two files (`id_rsa` and `id_rsa.pub`) inside `~/.ssh/`.

#### Related topics

- [SCP setup on page 2653](#)
- [Copying public SSH key to each destination host on page 2654](#)

## Copying public SSH key to each destination host

Transfer public key to each destination host as the transfer user:

Easy key copy:

```
ssh-copy-id [-i [identity_file]] [user@]machine
```

Manual steps to copy keys:

```
> scp ~/.ssh/id_rsa.pub destHost:~ (enter password)
```

Create an `authorized_keys` file on each destination host:

```
> ssh destHost (enter password)
> cat id_rsa.pub >> .ssh/authorized_keys
```

If the `.ssh` directory does not exist, create it with 700 privileges (`mkdir .ssh; chmod 700 .ssh`):

```
> chmod 700 .ssh/authorized_keys
```

## Related topics

- [Generating SSH key on source host on page 2654](#)
- [SCP setup on page 2653](#)

## Configuring the SSH daemon on each destination host

Some configuration of the SSH daemon may be required on the destination host. (Because this is not always the case, see [Validating correct SSH configuration on page 2655](#) and test the changes made to this point. If the tests fail, proceed with this step and then try testing again.) Typically, this is done by editing the `/etc/ssh/sshd_config` file (root access needed). To verify correct configuration, see that the following attributes are set (not commented):

```
RSAAuthentication yes
PubkeyAuthentication yes
```

If configuration changes were required, the SSH daemon will need to be restarted (root access needed):

```
> /etc/init.d/sshd restart
```

## Related topics

- [SCP setup on page 2653](#)

## Validating correct SSH configuration

If all is properly configured, the following command issued on the *source* host should succeed and not prompt for a password:

```
> scp destHost:/etc/motd /tmp
```

**i** If this is your first time accessing *destination* from *source*, it may ask you if you want to add the fingerprint to a file of known hosts. If you specify yes, this message should no longer appear and should not interfere with scp copying via TORQUE. Also, it is important that the full hostname appear in the `known_hosts` file. To do this, use the full hostname for *destHost*, as in `machine.domain.org` instead of just `machine`.

## Related topics

- [SCP setup on page 2653](#)

## Enabling bi-directional SCP access

The preceding steps allow *source* access to *destination* without prompting for a password. The reverse, however, is not true. Repeat the steps, but this time using the *destination* as the *source*, etc. to enable bi-directional SCP access (i.e. *source* can send to *destination* and *destination* can send to *source* without password prompts.)

## Related topics

- [SCP setup on page 2653](#)

## Compiling TORQUE to support SCP

**i** In TORQUE 2.1 and later, SCP is the default remote copy protocol. These instructions are only necessary for earlier versions.

TORQUE must be re-configured (and then rebuilt) to use SCP by passing in the `--with-scp` flag to the configure script:

```
> ./configure --prefix=xxx --with-scp
> make
```

**i** If special SCP flags are required in your local setup, these can be specified using the `$rcpcmd` parameter.

## Related topics

- [SCP setup on page 2653](#)

## Troubleshooting

If, after following all of the instructions in this section (see [SCP setup on page 2653](#)), TORQUE is still having problems transferring data with SCP, set the `PBSDEBUG` environment variable and restart the `pbs_mom` for details about copying. Also check the MOM log files for more details.

## Related topics

- [SCP setup on page 2653](#)

## NFS and other networked filesystems

When a batch job starts, its `stdin` file (if specified) is copied from the submission directory on the remote submission host. This file is placed in the `$PBSMOMHOME` directory on the mother superior node (i.e., `/usr/spool/PBS/spool`). As the job runs, `stdout` and `stderr` files are generated and placed in this directory using the naming convention `$JOBID.OU` and `$JOBID.ER`.

When the job completes, the MOM copies the files into the directory from which the job was submitted. By default, this file copying will be accomplished using a remote copy facility such as `rcp` or `scp`.

If a shared file system such as NFS, DFS, or AFS is available, a site can specify that the MOM should take advantage of this by specifying the `$usecp` directive inside the MOM configuration file (located in the `$PBSMOMHOME/mom_priv` directory) using the following format:

```
$usecp <HOST>:<SRCDIR> <DSTDIR>
```

<HOST> can be specified with a leading wildcard (\*) character. The following example demonstrates this directive:

```
mom_priv/config
# /home is NFS mounted on all hosts
$usecp */:/home /home
# submission hosts in domain fte.com should map '/data' directory on submit host to
# '/usr/local/data' on compute host
$usecp *.fte.com:/data /usr/local/data
```

If for any reason the MOM daemon is unable to copy the output or error files to the submission directory, these files are instead copied to the `undelivered` directory also located in `$PBSMOMHOME`.

### Related topics

- [Configuring data management on page 2653](#)

## File stage-in/stage-out

File staging requirements are specified using the `stagein` and `stageout` directives of the [qsub](#) command. Stagein requests occur before the job starts execution, while stageout requests happen after a job completes.

On completion of the job, all staged-in and staged-out files are removed from the execution system. The `file_list` is in the form `local_file@hostname:remote_file[,...]` regardless of the direction of the copy. The name `local_file` is the name of the file on the system where the job executed. It may be an absolute path or relative to the home directory of the user. The name `remote_file` is the destination name on the host specified by `hostname`. The name may be absolute or relative to the user's home directory on the destination host. The use of wildcards in the file name is not recommended.

The file names map to a remote copy program (`rcp/scp/cp`, depending on configuration) called on the execution system in the following manner:

For stagein: `rcp/scp hostname:remote_file local_file`

For stageout: `rcp/scp local_file hostname:remote_file`

### Examples

```
# stage /home/john/input_source.txt from node13.fsc to /home/john/input_
destination.txt on master compute node
> qsub -l nodes=1,walltime=100 -W stagein=input_
source.txt@node13.fsc:/home/john/input_destination.txt
```

```
# stage /home/bill/output_source.txt on master compute node to /tmp/output_
destination.txt on node15.fsc
> qsub -l nodes=1,walltime=100 -W stageout=/tmp/output_
source.txt@node15.fsc:/home/bill/output_destination.txt
```

```
$ fortune >xxx;echo cat xxx|qsub -W stagein=xxx@`hostname`:xxx
199.myhost.mydomain
$ cat STDIN*199
Anyone who has had a bull by the tail knows five or six more things
than someone who hasn't.
-- Mark Twain
```

## Related topics

- [Configuring data management on page 2653](#)

# MPI (Message Passing Interface) support

A message passing library is used by parallel jobs to augment communication between the tasks distributed across the cluster. TORQUE can run with any message passing library and provides limited integration with some [MPI](#) libraries.

For more information, see these topics:

- [MPICH on page 2658](#)
- [Open MPI on page 2659](#)

## MPICH

One of the most popular MPI libraries is [MPICH](#) available from [Argonne National Lab](#). If using this release, you may want to consider also using the [mpiexec](#) tool for launching MPI applications. Support for `mpiexec` has been integrated into TORQUE.

## MPIExec Overview

`mpiexec` is a replacement program for the script `mpirun`, which is part of the `mpich` package. It is used to initialize a parallel job from within a PBS batch or interactive environment. `mpiexec` uses the task manager library of PBS to spawn copies of the executable on the nodes in a PBS allocation.

Reasons to use `mpiexec` rather than a script (`mpirun`) or an external daemon (`mpd`):

- Starting tasks with the task manager (TM) interface is much faster than invoking a separate `rsh` \* once for each process.
- Resources used by the spawned processes are accounted correctly with `mpiexec`, and reported in the PBS logs, because all the processes of a parallel job remain under the control of PBS, unlike when using `mpirun`-like scripts.
- Tasks that exceed their assigned limits of CPU time, wallclock time, memory usage, or disk space are killed cleanly by PBS. It is quite hard for processes to escape control of the resource manager when using `mpiexec`.
- You can use `mpiexec` to enforce a security policy. If all jobs are forced to spawn using `mpiexec` and the PBS execution environment, it is not necessary to enable `rsh` or `ssh` access to the compute nodes in the cluster.

For more information, see the [mpiexec](#) homepage.

## MPIExec Troubleshooting

Although problems with `mpiexec` are rare, if issues do occur, the following steps may be useful:

- Determine current version using `mpiexec --version` and review the [change log](#) available on the [MPI homepage](#) to determine if the reported issue has already been corrected.
- Send email to the mpiexec mailing list at [mpiexec@osc.edu](mailto:mpiexec@osc.edu).
- Browse the mpiexec user list [archives](#) for similar problems and resolutions.
- Read the FAQ contained in the README file and the mpiexec man pages contained within the mpiexec distribution.
- Increase the logging of mpiexec operation with `mpiexec --verbose` (reports messages to `stderr`).
- Increase logging of the master and slave resource manager execution daemons associated with the job (with TORQUE, use `$loglevel` to 5 or higher in `$TORQUEROOT/mom_priv/config` and look for 'tm' messages after associated `join` job messages).
- Use `tracejob` (included with TORQUE) or `qtracejob` (included with OSC's pbstools package) to isolate failures within the cluster.
- If the message 'exec: Error: get\_hosts: pbs\_connect: Access from host not allowed, or unknown host' appears, this indicates that mpiexec cannot communicate with the `pbs_server` daemon. In most cases, this indicates that the `$TORQUEROOT/server_name` file points to the wrong server or the node cannot resolve the server's name. The [qstat](#) command can be run on the node to test this.

## General MPI Troubleshooting

When using MPICH, some sites have issues with orphaned MPI child processes remaining on the system after the master MPI process has been terminated. To address this, TORQUE epilogue scripts can be created that properly clean up the orphaned processes (see [Appendix G: Prologue and epilogue scripts on page 2817](#)).

Related topics

- [MPI \(Message Passing Interface\) support on page 2658](#)

## Open MPI

[Open MPI](#) is a new MPI implementation that combines technologies from multiple projects to create the best possible library. It supports the TM interface for integration with TORQUE. More information is available in the [FAQ](#).

Related topics

- [MPI \(Message Passing Interface\) support on page 2658](#)

# Resources

A primary task of any resource manager is to monitor the state, health, configuration, and utilization of managed resources. TORQUE is specifically designed to monitor compute hosts for use in a batch environment. TORQUE is not designed to monitor non-compute host resources such as software licenses, networks, file systems, and so forth, although these resources can be integrated into the cluster using some scheduling systems.

With regard to monitoring compute nodes, TORQUE reports about a number of attributes broken into three major categories:

- [Configuration on page 2660](#)
- [Utilization on page 2661](#)
- [Node states on page 2661](#)

## Configuration

Configuration includes both detected hardware configuration and specified batch attributes.

Attribute	Description	Details
<b>Architecture (arch)</b>	operating system of the node	The value reported is a derivative of the operating system installed.
<b>Node Features (properties)</b>	arbitrary string attributes associated with the node	No node features are specified by default. If required, they are set using the <code>nodes</code> file located in the <code>TORQUE_HOME/server_priv</code> directory. They may specify any string and are most commonly used to allow users to request certain subsets of nodes when submitting jobs.
<b>Local Disk (size)</b>	configured local disk	By default, local disk space is not monitored. If the MOM configuration <a href="#">size [fs=&lt;FS&gt;]</a> parameter is set, TORQUE will report, in kilobytes, configured disk space within the specified directory.
<b>Memory (physmem)</b>	local memory/RAM	Local memory/RAM is monitored and reported in kilobytes.

Attribute	Description	Details
<b>Processors (ncpus/np)</b>	real/virtual processors	The number of processors detected by TORQUE is reported via the <i>ncpus</i> attribute. However, for scheduling purposes, other factors are taken into account. In its default configuration, TORQUE operates in "dedicated" mode with each node possessing a single virtual processor. In dedicated mode, each job task will consume one virtual processor and TORQUE will accept workload on each node until all virtual processors on that node are in use. While the number of virtual processors per node defaults to 1, this may be configured using the <i>nodes</i> file located in the <code>TORQUE_HOME/server_priv</code> directory. An alternative to dedicated mode is "timeshared" mode. If TORQUE's time-shared mode is enabled, TORQUE will accept additional workload on each node until the node's <i>maxload</i> limit is reached.
<b>Swap (totmem)</b>	virtual memory/Swap	Virtual memory/Swap is monitored and reported in kilobytes.

## Utilization

Utilization includes information regarding the amount of node resources currently in use as well as information about who or what is consuming it.

Attribute	Description	Details
<b>Disk (size)</b>	local disk availability	By default, local disk space is not monitored. If the MOM configuration <a href="#">size [fs=&lt;FS&gt;]</a> parameter is set, TORQUE will report configured and currently available disk space within the specified directory in kilobytes.
<b>Memory (availmem)</b>	real memory/RAM	Available real memory/RAM is monitored and reported in kilobytes.
<b>Network (netload)</b>	local network adapter usage	Reports total number of bytes transferred in or out by the network adapter.
<b>Processor Utilization (loadave)</b>	node's cpu load average	Reports the node's 1 minute bsd load average.

## Node states

State information includes administrative status, general node health information, and general usage status.

Attribute	Description	Details
<b>Idle Time (idletime)</b>	time since local keyboard/mouse activity has been detected	Time in seconds since local keyboard/mouse activity has been detected.
<b>State (state)</b>	monitored/admin node state	<p>A node can be in one or more of the following states:</p> <ul style="list-style-type: none"> <li>• <i>busy</i> - node is full and will not accept additional work</li> <li>• <i>down</i> - node is failing to report, is detecting local failures with node</li> <li>• <i>free</i> - node is ready to accept additional work</li> <li>• <i>job-exclusive</i> - all available virtual processors are assigned to jobs</li> <li>• <i>job-sharing</i> - node has been allocated to run multiple shared jobs and will remain in this state until jobs are complete</li> <li>• <i>offline</i> - node has been instructed by an admin to no longer accept work</li> <li>• <i>reserve</i> - node has been reserved by the server</li> <li>• <i>time-shared</i> - node always allows multiple jobs to run concurrently</li> <li>• <i>unknown</i> - node has not been detected</li> </ul>

## Accounting records

TORQUE maintains accounting records for batch jobs in the following directory:

```
$TORQUEROOT/server_priv/accounting/<TIMESTAMP>
```

\$TORQUEROOT defaults to /usr/spool/PBS and <TIMESTAMP> is in the format: YYYYMMDD.

These records include events, time stamps, and information on resources requested and used.

Records for four different event types are produced and are described in the following table:

Record marker	Record type	Description
<b>A</b>	abort	Job has been aborted by the server
<b>C</b>	checkpoint	Job has been checkpointed and held
<b>D</b>	delete	Job has been deleted

Record marker	Record type	Description
<b>E</b>	exit	Job has exited (either successfully or unsuccessfully)
<b>Q</b>	queue	Job has been submitted/queued
<b>R</b>	rerun	Attempt to rerun the job has been made
<b>S</b>	start	Attempt to start the job has been made (if the job fails to properly start, it may have multiple job start records)
<b>T</b>	restart	Attempt to restart the job (from checkpoint) has been made (if the job fails to properly start, it may have multiple job start records)

## Accounting Variables

The following table offers accounting variable descriptions. Descriptions for accounting variables not indicated in the table, particularly those prefixed with **Resources\_List**, are available at [Job submission on page 2583](#).

Variable	Description
<b>ctime</b>	Time job was created
<b>etime</b>	Time job became eligible to run
<b>qtime</b>	Time job was queued
<b>start</b>	Time job started to run

A sample record in this file can look like the following:

```
06/06/2005 14:04:25;D;408.ign1.zeta2.org;requestor=guest@ign1.zeta2.org
06/06/2005 14:04:35;Q;409.ign1.zeta2.org;queue=batch
06/06/2005 14:04:44;Q;410.ign1.zeta2.org;queue=batch
06/06/2005 14:06:06;S;407.ign1.zeta2.org;user=guest group=guest jobname=STDIN
queue=batch ctime=1118087915 qtime=1118087915 etime=1118087915 start=1118088366 exec_
host=ign1.zeta2.org/0 Resource_List.neednodes=ign1.zeta2.org Resource_List.nodect=1
Resource_List.nodes=1 Resource_List.walltime=00:16:40
06/06/2005 14:07:17;D;407.ign1.zeta2.org;requestor=guest@ign1.zeta2.org
06/06/2005 14:07:17;E;407.ign1.zeta2.org;user=guest group=guest jobname=STDIN
queue=batch ctime=1118087915 qtime=1118087915 etime=1118087915 start=1118088366 exec_
host=ign1.zeta2.org/0 Resource_List.nodect=1 Resource_List.nodes=1 Resource_
List.walltime=00:16:40 session=6365 end=1118088437 Exit_status=271 resources_
used.cput=00:00:00 resources_used.mem=3068kb resources_used.vmem=16080kb resources_
used.walltime=00:01:11
```

**i** The value of `Resource_List.*` is the amount of resources requested, and the value of `resources_used.*` is the amount of resources actually used.

## Job logging

New in TORQUE 2.5.3 is the ability to log job information for completed jobs. The information stored in the log file is the same information produced with the command `qstat -f`. The log file data is stored using an XML format. Data can be extracted from the log using the utility `showjobs` found in the `contrib/` directory of the TORQUE source tree. Custom scripts that can parse the XML data can also be used.

For details about job logging, see these topics:

- [Job log location and name on page 2664](#)
- [Enabling job logs on page 2664](#)

### Job log location and name

When job logging is enabled (see [Enabling job logs on page 2664](#)), the job log is kept at `$TORQUE_HOME/job_logs`. The naming convention for the job log is the same as for the server log or MOM log. The log name is created from the current year/month/day.

For example, if today's date is 26 October, 2010 the log file is named 20101026.

A new log file is created each new day that data is written to the log.

Related topics

- [Enabling job logs on page 2664](#)
- [Job logging on page 2664](#)

### Enabling job logs

There are five new server parameters used to enable job logging. These parameters control what information is stored in the log and manage the log files.

Parameter	Description
<b>record_job_info</b>	This must be set to true in order for job logging to be enabled. If not set to true, the remaining server parameters are ignored.

Parameter	Description
<b>record_job_script</b>	If set to true, this adds the contents of the script executed by a job to the log.
<b>job_log_file_max_size</b>	This specifies a soft limit (in kilobytes) for the job log's maximum size. The file size is checked every five minutes and if the <i>current day</i> file size is greater than or equal to this value, it is rolled from <i>&lt;filename&gt;</i> to <i>&lt;filename.1&gt;</i> and a new empty log is opened. If the current day file size exceeds the maximum size a second time, the <i>&lt;filename.1&gt;</i> log file is rolled to <i>&lt;filename.2&gt;</i> , the current log is rolled to <i>&lt;filename.1&gt;</i> , and a new empty log is opened. Each new log causes all other logs to roll to an extension that is one greater than its current number. Any value less than 0 is ignored by pbs_server (meaning the log will not be rolled).
<b>job_log_file_roll_depth</b>	This sets the maximum number of new log files that are kept in a day if the <a href="#">job_log_file_max_size</a> parameter is set. For example, if the roll depth is set to 3, no file can roll higher than <i>&lt;filename.3&gt;</i> . If a file is already at the specified depth, such as <i>&lt;filename.3&gt;</i> , the file is deleted so it can be replaced by the incoming file roll, <i>&lt;filename.2&gt;</i> .
<b>job_log_keep_days</b>	This maintains logs for the number of days designated. If set to 4, any log file older than 4 days old is deleted.

#### Related topics

- [Job log location and name](#) on page 2664
- [Job logging](#) on page 2664

## Troubleshooting

There are a few general strategies that can be followed to determine the cause of unexpected behavior. These are a few of the tools available to help determine where problems occur. See these topics for details:

- [Host resolution](#) on page 2666
- [Firewall configuration](#) on page 2666
- [TORQUE log files](#) on page 2666
- [Using "tracejob" to locate job failures](#) on page 2668
- [Using GDB to locate job failures](#) on page 2670
- [Other diagnostic options](#) on page 2670
- [Stuck jobs](#) on page 2671
- [Frequently asked questions \(FAQ\)](#) on page 2672

- [Compute node health check on page 2677](#)
- [Debugging on page 2679](#)

## Host resolution

The TORQUE server host must be able to perform both forward and reverse name lookup on itself and on all compute nodes. Likewise, each compute node must be able to perform forward and reverse name lookup on itself, the TORQUE server host, and all other compute nodes. In many cases, name resolution is handled by configuring the node's `/etc/hosts` file although *DNS* and *NIS* services may also be used. Commands such as `nslookup` or `dig` can be used to verify proper host resolution.

**i** Invalid host resolution may exhibit itself with compute nodes reporting as down within the output of `pbsnodes-a` and with failure of the `momctl -d3` command.

### Related topics

- [Troubleshooting on page 2665](#)

## Firewall configuration

Be sure that, if you have firewalls running on the server or node machines, you allow connections on the appropriate ports for each machine. TORQUE `pbs_mom` daemons use UDP ports 1023 and below if privileged ports are configured (privileged ports is the default). The `pbs_server` and `pbs_mom` daemons use TCP and UDP ports 15001-15004 by default.

Firewall based issues are often associated with server to MOM communication failures and messages such as 'premature end of message' in the log files.

Also, the `tcpdump` program can be used to verify the correct network packets are being sent.

### Related topics

- [Troubleshooting on page 2665](#)

## TORQUE log files

### pbs\_server and pbs\_mom log files

The `pbs_server` keeps a daily log of all activity in the `TORQUE_HOME/server_logs` directory. The `pbs_mom` also keeps a daily log of all activity in the `TORQUE_HOME/mom_logs/` directory. These logs contain information on communication between server and MOM as well as information on jobs as they enter the queue and as they are dispatched, run, and terminated. These logs can be very helpful in determining general job failures. For MOM logs, the verbosity of the logging can be adjusted by setting the `$loglevel` parameter in the `mom_priv/config` file. For server logs, the verbosity of the logging can be adjusted by setting the server `log_level` attribute in `qmgr`.

For both `pbs_mom` and `pbs_server` daemons, the log verbosity level can also be adjusted by setting the environment variable **PBSLOGLEVEL** to a value between 0 and 7. Further, to dynamically change the log level of a running daemon, use the `SIGUSR1` and `SIGUSR2` signals to increase and decrease the active loglevel by one. Signals are sent to a process using the `kill` command.

For example, `kill -USR1 `pgrep pbs_mom`` would raise the log level up by one.

The current loglevel for `pbs_mom` can be displayed with the command `momctl -d3`.

## trqauthd log files

As of TORQUE 4.1.3, `trqauthd` logs its events in the `$TORQUE_HOME/client_logs` directory. It names the log files in the format `<YYYYMMDD>`, creating a new log daily as events occur.

**i** You might see some peculiar behavior if you mount the `client_logs` directory for shared access via network-attached storage.

When `trqauthd` first gets access on a particular day, it writes an "open" message to the day's log file. It also writes a "close" message to the last log file it accessed prior to that, which is usually the previous day's log file, but not always. For example, if it is Monday and no client commands were executed over the weekend, `trqauthd` writes the "close" message to Friday's file.

Since the various `trqauthd` binaries on the submit hosts (and potentially, the compute nodes) each write an "open" and "close" message on the first access of a new day, you'll see multiple (seemingly random) accesses when you have a shared log.

The `trqauthd` records the following events along with the date and time of the occurrence:

- When `trqauthd` successfully starts. It logs the event with the IP address and port.
- When a user successfully authenticates with `trqauthd`.
- When a user fails to authenticate with `trqauthd`.
- When `trqauthd` encounters any unexpected errors.

### Example 4-26: trqauthd logging sample

```
2012-10-05 15:05:51.8404 Log opened
2012-10-05 15:05:51.8405 TORQUE authd daemon started and listening on IP:port
101.0.1.0:12345
2012-10-10 14:48:05.5688 User hfrye at IP:port abc:12345 logged in
```

### Related topics

- [Troubleshooting on page 2665](#)

## Using "tracejob" to locate job failures

### Overview

The *tracejob* utility extracts job status and job events from accounting records, MOM log files, server log files, and scheduler log files. Using it can help identify where, how, a why a job failed. This tool takes a job id as a parameter as well as arguments to specify which logs to search, how far into the past to search, and other conditions.

### Syntax

```
tracejob [-a|s|l|m|q|v|z] [-c count] [-w size] [-p path] [ -n <DAYS>] [-f
filter_type] <JOBID>
```

```
-p : path to PBS_SERVER_HOME
-w : number of columns of your terminal
-n : number of days in the past to look for job(s) [default 1]
-f : filter out types of log entries, multiple -f's can be specified
    error, system, admin, job, job_usage, security, sched, debug,
    debug2, or absolute numeric hex equivalent
-z : toggle filtering excessive messages
-c : what message count is considered excessive
-a : don't use accounting log files
-s : don't use server log files
-l : don't use scheduler log files
-m : don't use MOM log files
-q : quiet mode - hide all error messages
-v : verbose mode - show more error messages
```

## Example

```

> tracejob -n 10 1131
Job: 1131.icluster.org

03/02/2005 17:58:28 S enqueueing into batch, state 1 hop 1
03/02/2005 17:58:28 S Job Queued at request of dev@icluster.org, owner =
dev@icluster.org, job name = STDIN, queue = batch
queue=batch
03/02/2005 17:58:28 A
03/02/2005 17:58:41 S Job Run at request of dev@icluster.org
03/02/2005 17:58:41 M evaluating limits for job
03/02/2005 17:58:41 M phase 2 of job launch successfully completed
03/02/2005 17:58:41 M saving task (TMomFinalizeJob3)
03/02/2005 17:58:41 M job successfully started
03/02/2005 17:58:41 M job 1131.koa.icluster.org reported successful start on 1 node
(s)
03/02/2005 17:58:41 A user=dev group=dev jobname=STDIN queue=batch ctime=1109811508

qtime=1109811508 etime=1109811508 start=1109811521
exec_host=icluster.org/0 Resource_List.neednodes=1 Resource_
List.nodect=1
Resource_List.nodes=1 Resource_List.walltime=00:01:40
03/02/2005 18:02:11 M walltime_210 exceeded limit 100
03/02/2005 18:02:11 M kill_job
03/02/2005 18:02:11 M kill_job found a task to kill
03/02/2005 18:02:11 M sending signal 15 to task
03/02/2005 18:02:11 M kill_task: killing pid 14060 task 1 with sig 15
03/02/2005 18:02:11 M kill_task: killing pid 14061 task 1 with sig 15
03/02/2005 18:02:11 M kill_task: killing pid 14063 task 1 with sig 15
03/02/2005 18:02:11 M kill_job done
03/02/2005 18:04:11 M kill_job
03/02/2005 18:04:11 M kill_job found a task to kill
03/02/2005 18:04:11 M sending signal 15 to task
03/02/2005 18:06:27 M kill_job
03/02/2005 18:06:27 M kill_job done
03/02/2005 18:06:27 M performing job clean-up
03/02/2005 18:06:27 A user=dev group=dev jobname=STDIN queue=batch ctime=1109811508

qtime=1109811508 etime=1109811508 start=1109811521
exec_host=icluster.org/0 Resource_List.neednodes=1 Resource_
List.nodect=1
Resource_List.nodes=1 Resource_List.walltime=00:01:40
session=14060

end=1109811987 Exit_status=265 resources_used.cput=00:00:00
resources_used.mem=3544kb resources_used.vmem=10632kb
resources_used.walltime=00:07:46
...

```

**i** The `tracejob` command operates by searching the `pbs_server` accounting records and the `pbs_server`, `MOM`, and scheduler logs. To function properly, it must be run on a node and as a user which can access these files. By default, these files are all accessible by the user `root` and only available on the cluster management node. In particular, the files required by `tracejob` are located in the following directories:

```
TORQUE_HOME/server_priv/accounting
```

```
TORQUE_HOME/server_logs
```

```
TORQUE_HOME/mom_logs
```

```
TORQUE_HOME/sched_logs
```

**i** `tracejob` may only be used on systems where these files are made available. Non-root users may be able to use this command if the permissions on these directories or files are changed appropriately.

**i** The value of `Resource_List.*` is the amount of resources requested, and the value of `resources_used.*` is the amount of resources actually used.

### Related topics

- [Troubleshooting on page 2665](#)

## Using GDB to locate job failures

If either the `pbs_mom` or `pbs_server` fail unexpectedly (and the log files contain no information on the failure) `gdb` can be used to determine whether or not the program is crashing. To start `pbs_mom` or `pbs_server` under [GDB](#) export the environment variable `PBSDEBUG=yes` and start the program (i.e., `gdb pbs_mom` and then issue the `run` subcommand at the `gdb` prompt).

`GDB` may run for some time until a failure occurs and at which point, a message will be printed to the screen and a `gdb` prompt again made available. If this occurs, use the `gdb where` subcommand to determine the exact location in the code. The information provided may be adequate to allow local diagnosis and correction. If not, this output may be sent to the mailing list or to [help](#) for further assistance.

**i** See the `PBSCOREDUMP` parameter for enabling creation of core files (see [Using "tracejob" to locate job failures on page 2668](#)).

### Related topics

- [Troubleshooting on page 2665](#)

## Other diagnostic options

When `PBSDEBUG` is set, some client commands will print additional diagnostic information.

```
$ export PBSDEBUG=yes
$ cmd
```

To debug different kinds of problems, it can be useful to see where in the code time is being spent. This is called profiling and there is a Linux utility "gprof" that will output a listing of routines and the amount of time spent in these routines. This does require that the code be compiled with special options to instrument the code and to produce a file, `gmon.out`, that will be written at the end of program execution.

The following listing shows how to build TORQUE with profiling enabled. Notice that the output file for `pbs_mom` will end up in the `mom_priv` directory because its startup code changes the default directory to this location.

```
# ./configure "CFLAGS=-pg -lgcov -fPIC"
# make -j5
# make install
# pbs_mom ... do some stuff for a while ...
# momctl -s
# cd /var/spool/torque/mom_priv
# gprof -b `which pbs_mom` gmon.out |less
#
```

Another way to see areas where a program is spending most of its time is with the `valgrind` program. The advantage of using `valgrind` is that the programs do not have to be specially compiled.

```
# valgrind --tool=callgrind pbs_mom
```

#### Related topics

- [Troubleshooting on page 2665](#)

## Stuck jobs

If a job gets stuck in TORQUE, try these suggestions to resolve the issue:

- Use the [qdel](#) command to cancel the job.
- Force the MOM to send an obituary of the job ID to the server.

```
> qsig -s 0 <JOBID>
```

- You can try clearing the stale jobs by using the [momctl](#) command on the compute nodes where the jobs are still listed.

```
> momctl -c 58925 -h compute-5-20
```

- Setting the [qmgr](#) server setting `mom_job_sync` to *True* might help prevent jobs from hanging.

```
> qmgr -c "set server mom_job_sync = True"
```

To check and see if this is already set, use:

```
> qmgr -c "p s"
```

- If the suggestions above cannot remove the stuck job, you can try [qdel](#) `-p`. However, since the `-p` option purges all information generated by the job, this is not a recommended option unless the above suggestions fail to remove the stuck job.

```
> qdel -p <JOBID>
```

- The last suggestion for removing stuck jobs from compute nodes is to restart the `pbs_mom`.

For additional troubleshooting, run a `tracejob` on one of the stuck jobs. You can then create an [online support ticket](#) with the full server log for the time period displayed in the trace job.

#### Related topics

- [Troubleshooting on page 2665](#)

## Frequently asked questions (FAQ)

- [Cannot connect to server: error=15034 on page 2672](#)
- [Deleting 'stuck' jobs on page 2672](#)
- [Which user must run TORQUE? on page 2673](#)
- [Scheduler cannot run jobs - rc: 15003 on page 2673](#)
- [PBS\\_Server: pbsd\\_init, Unable to read server database on page 2673](#)
- [qsub will not allow the submission of jobs requesting many processors on page 2675](#)
- [qsub reports 'Bad UID for job execution' on page 2675](#)
- [Why does my job keep bouncing from running to queued? on page 2675](#)
- [How do I use PVM with TORQUE? on page 2676](#)
- [My build fails attempting to use the TCL library on page 2676](#)
- [My job will not start, failing with the message 'cannot send job to mom, state=PRERUN' on page 2676](#)
- [How do I determine what version of TORQUE I am using? on page 2676](#)
- [How do I resolve autogen.sh errors that contain "error: possibly undefined macro: AC\\_MSG\\_ERROR"? on page 2676](#)
- [How do I resolve compile errors with libssl or libcrypto for TORQUE 4.0 on Ubuntu 10.04? on page 2677](#)
- [Why are there so many error messages in the client logs \(trqauthd logs\) when I don't notice client commands failing? on page 2677](#)

### Cannot connect to server: error=15034

This error occurs in TORQUE clients (or their APIs) because TORQUE cannot find the `server_name` file and/or the `PBS_DEFAULT` environment variable is not set. The `server_name` file or `PBS_DEFAULT` variable indicate the `pbs_server`'s hostname that the client tools should communicate with. The `server_name` file is usually located in TORQUE's local state directory. Make sure the file exists, has proper permissions, and that the version of TORQUE you are running was built with the proper directory settings. Alternatively you can set the `PBS_DEFAULT` environment variable. Restart TORQUE daemons if you make changes to these settings.

### Deleting 'stuck' jobs

To manually delete a "stale" job which has no process, and for which the mother superior is still alive, sending a sig 0 with `qsig` will often cause MOM to realize the job is stale and issue the proper JobObit notice. Failing that, use `momctl -c` to forcefully cause MOM to purge the job. The following process should never be necessary:

- Shut down the MOM on the mother superior node.
- Delete all files and directories related to the job from `TORQUE_HOME/mom_priv/jobs`.
- Restart the MOM on the mother superior node.

If the mother superior MOM has been lost and cannot be recovered (i.e. hardware or disk failure), a job running on that node can be purged from the output of [qstat](#) using the [qdel on page 2720](#) `-p` command or can be removed manually using the following steps:

### To remove job X

1. Shut down `pbs_server`.

```
> qterm
```

2. Remove job spool files.

```
> rm TORQUE_HOME/server_priv/jobs/X.SC TORQUE_HOME/server_priv/jobs/X.JB
```

3. Restart `pbs_server`

```
> pbs_server
```

## Which user must run TORQUE?

TORQUE (`pbs_server` & `pbs_mom`) must be started by a user with root privileges.

## Scheduler cannot run jobs - rc: 15003

For a scheduler, such as [Moab](#) or [Maui](#), to control jobs with TORQUE, the scheduler needs to be run by a user in the server operators / managers list (see [qmgr](#)). The default for the server operators / managers list is `root@localhost`. For TORQUE to be used in a grid setting with Silver, the scheduler needs to be run as `root`.

## PBS\_Server: pbsd\_init, Unable to read server database

If this message is displayed upon starting `pbs_server` it means that the local database cannot be read. This can be for several reasons. The most likely is a version mismatch. Most versions of TORQUE can read each other's databases. However, there are a few incompatibilities between OpenPBS and TORQUE. Because of enhancements to TORQUE, it cannot read the job database of an OpenPBS server (job structure sizes have been altered to increase functionality). Also, a compiled in 32-bit mode cannot read a database generated by a 64-bit `pbs_server` and vice versa.

## To reconstruct a database (excluding the job database)

1. First, print out the old data with this command:

```
%> qmgr -c "p s"
#
# Create queues and set their attributes.
#
#
# Create and define queue batch
# create queue batch
set queue batch queue_type = Execution
set queue batch acl_host_enable = False
set queue batch resources_max.nodect = 6
set queue batch resources_default.nodes = 1
set queue batch resources_default.walltime = 01:00:00
set queue batch resources_available.nodect = 18
set queue batch enabled = True
set queue batch started = True
#
# Set server attributes.
#
set server scheduling = True
set server managers = griduser@oahu.icluster.org
set server managers += scott@*.icluster.org
set server managers += wightman@*.icluster.org
set server operators = griduser@oahu.icluster.org
set server operators += scott@*.icluster.org
set server operators += wightman@*.icluster.org
set server default_queue = batch
set server log_events = 511
set server mail_from = adm
set server resources_available.nodect = 80
set server node_ping_rate = 300
set server node_check_rate = 600
set server tcp_timeout = 6
```

2. Copy this information somewhere.
3. Restart pbs\_server with the following command:

```
> pbs_server -t create
```

4. When you are prompted to overwrite the previous database, enter *y*, then enter the data exported by the `qmgr` command as in this example:

```
> cat data | qmgr
```

5. Restart pbs\_server without the flags:

```
> qterm
> pbs_server
```

This will reinitialize the database to the current version.

**i** Reinitializing the server database will reset the next jobid to 1

## qsub will not allow the submission of jobs requesting many processors

TORQUE's definition of a node is context sensitive and can appear inconsistent. The `qsub -l nodes=<X>` expression can at times indicate a request for X processors and other time be interpreted as a request for X nodes. While `qsub` allows multiple interpretations of the keyword nodes, aspects of the TORQUE server's logic are not so flexible. Consequently, if a job is using `-l nodes` to specify processor count and the requested number of processors exceeds the available number of physical nodes, the server daemon will reject the job.

To get around this issue, the server can be told it has an inflated number of nodes using the `resources_available` attribute. To take effect, this attribute should be set on both the server and the associated queue as in the example below. (See [resources\\_available](#) for more information.)

```
> qmgr
Qmgr: set server resources_available.nodect=2048
Qmgr: set queue batch resources_available.nodect=2048
```

**i** The `pbs_server` daemon will need to be restarted before these changes will take effect.

## qsub reports 'Bad UID for job execution'

```
[guest@login2]$ qsub test.job
qsub: Bad UID for job execution
```

Job submission hosts must be explicitly specified within TORQUE or enabled via RCmd security mechanisms in order to be trusted. In the example above, the host 'login2' is not configured to be trusted. This process is documented in [Configuring job submission hosts on page 2566](#).

## Why does my job keep bouncing from running to queued?

There are several reasons why a job will fail to start. Do you see any errors in the MOM logs? Be sure to increase the loglevel on MOM if you don't see anything. Also be sure TORQUE is configured with `--enable-syslog` and look in `/var/log/messages` (or wherever your syslog writes).

Also verify the following on all machines:

- DNS resolution works correctly with matching forward and reverse
- Time is synchronized across the head and compute nodes
- User accounts exist on all compute nodes
- User home directories can be mounted on all compute nodes
- Prologue scripts (if specified) exit with 0

If using a scheduler such as [Moab](#) or [Mauai](#), use a scheduler tool such as `checkjob` to identify job start issues.

## How do I use PVM with TORQUE?

- Start the master pvm on a compute node and then add the slaves
- mpiexec can be used to launch slaves using rsh or ssh (use `export PVM_RSH=/usr/bin/ssh` to use ssh)

**i** Access can be managed by rsh/ssh without passwords between the batch nodes, but denying it from anywhere else, including the interactive nodes. This can be done with xinetd and sshd configuration (root is allowed to ssh everywhere). This way, the pvm daemons can be started and killed from the job script.

The problem is that this setup allows the users to bypass the batch system by writing a job script that uses rsh/ssh to launch processes on the batch nodes. If there are relatively few users and they can more or less be trusted, this setup can work.

## My build fails attempting to use the TCL library

TORQUE builds can fail on TCL dependencies even if a version of TCL is available on the system. TCL is only utilized to support the xpbsmon client. If your site does not use this tool (most sites do not use xpbsmon), you can work around this failure by rerunning `configure` with the `--disable-gui` argument.

## My job will not start, failing with the message 'cannot send job to mom, state=PRERUN'

If a node crashes or other major system failures occur, it is possible that a job may be stuck in a corrupt state on a compute node. TORQUE 2.2.0 and higher automatically handle this when the `mom_job_sync` parameter is set via [qmgr](#) (the default). For earlier versions of TORQUE, set this parameter and restart the `pbs_mom` daemon.

This error can also occur if not enough free space is available on the partition that holds TORQUE.

## How do I determine what version of TORQUE I am using?

There are times when you want to find out what version of TORQUE you are using. An easy way to do this is to run the following command:

```
qmgr
> qmgr -c "p s" | grep pbs_ver
```

## How do I resolve autogen.sh errors that contain "error: possibly undefined macro: AC\_MSG\_ERROR"?

Verify the `pkg-config` package is installed.

## How do I resolve compile errors with libssl or libcrypto for TORQUE 4.0 on Ubuntu 10.04?

When compiling TORQUE 4.0 on Ubuntu 10.04 the following errors might occur:

```
libtool: link: gcc -Wall -pthread -g -D_LARGEFILE64_SOURCE -o .libs/trqauthd trq_auth_
daemon.o trq_main.o -ldl -lssl -lcrypto -L/home/adaptive/torques/torque-
4.0.0/src/lib/Libpbs/.libs /home/adaptive/torques/torque-
4.0.0/src/lib/Libpbs/.libs/libtorque.so -lpthread -lrt -pthread
/usr/bin/ld: cannot find -lssl
collect2: ld returned 1 exit status
make[3]: *** [trqauthd] Error 1

libtool: link: gcc -Wall -pthread -g -D_LARGEFILE64_SOURCE -o .libs/trqauthd trq_auth_
daemon.o trq_main.o -ldl -lssl -lcrypto -L/home/adaptive/torques/torque-
4.0.0/src/lib/Libpbs/.libs /home/adaptive/torques/torque-
4.0.0/src/lib/Libpbs/.libs/libtorque.so -lpthread -lrt -pthread
/usr/bin/ld: cannot find -lcrypto
collect2: ld returned 1 exit status
make[3]: *** [trqauthd] Error 1
```

To resolve the compile issue, use these commands:

```
> cd /usr/lib
> ln -s /lib/libcrypto.so.0.9.8 libcrypto.so
> ln -s /lib/libssl.so.0.9.8 libssl.so
```

## Why are there so many error messages in the client logs (trqauthd logs) when I don't notice client commands failing?

If a client makes a connection to the server and the trqauthd connection for that client command is authorized *before* the client's connection, the trqauthd connection is rejected. The connection is retried, but if all retry attempts are rejected, trqauthd logs a message indicating a failure. Some client commands then open a new connection to the server and try again. The client command fails only if all its retries fail.

### Related topics

- [Troubleshooting on page 2665](#)

## Compute node health check

TORQUE provides the ability to perform health checks on each compute node. If these checks fail, a failure message can be associated with the node and routed to the scheduler. Schedulers (such as [Moab](#)) can forward this information to administrators by way of scheduler triggers, make it available through scheduler diagnostic commands, and automatically mark the node down until the issue is resolved. (See the RMMMSGIGNORE parameter in [Appendix A: Moab Parameters on page 923](#) for more information.)

Additionally, Michael Jennings at LBNL has authored an open-source bash node health check script project. It offers an easy way to perform some of the most common node health checking tasks, such as verifying network and filesystem functionality. More information is available on the [project's page](#).

For more information about node health checks, see these topics:

- [Configuring MOMs to launch a health check on page 2678](#)
- [Creating the health check script on page 2678](#)
- [Adjusting node state based on the health check output on page 2679](#)
- [Example health check script on page 2679](#)

#### Related topics

- [Troubleshooting on page 2665](#)

## Configuring MOMs to launch a health check

The health check feature is configured via the `mom_priv/config` file using the parameters described below:

Parameter	Format	Default	Description
<code>\$node_check_script</code>	<STRING>	N/A	(Required) Specifies the fully qualified pathname of the health check script to run
<code>\$node_check_interval</code>	<INTEGER>	1	(Optional) Specifies the number of MOM intervals between health checks (by default, each MOM interval is 45 seconds long - this is controlled via the <a href="#">\$status_update_time on page 2798</a> node parameter. The integer may be followed by a list of event names (currently supported are <code>jobstart</code> and <code>jobend</code> ). (For more information, see <a href="#">pbs_mom</a> .)

#### Related topics

- [Compute node health check on page 2677](#)

## Creating the health check script

The health check script is executed directly by the `pbs_mom` daemon under the root user id. It must be accessible from the compute node and may be a script or compile executable program. It may make any needed system calls and execute any combination of system utilities but should not execute resource manager client commands. Also, as of TORQUE 1.0.1, the `pbs_mom` daemon blocks until the health check is completed and does not possess a built-in timeout. Consequently, it is advisable to keep the launch script execution time short and verify that the script will not block even under failure conditions.

If the script detects a failure, it should return the keyword **ERROR** to stdout followed by an error message. When a failure is detected, the **ERROR** keyword should be printed to stdout before any other data. The message (up to 1024 characters) immediately following the **ERROR** keyword must all be contained on the same line. The message is assigned to the node attribute 'message' of the associated node.

## Related topics

- [Compute node health check on page 2677](#)

## Adjusting node state based on the health check output

If the health check reports an error, the node attribute "message" is set to the error string returned. Cluster schedulers can be configured to adjust a given node's state based on this information. For example, by default, [Moab](#) sets a node's state to down if a node error message is detected. The node health script continues to run at the configured interval (see [Configuring MOMs to launch a health check on page 2678](#) for more information), and if it does not generate the error message again during one of its later executions, Moab picks that up at the beginning of its next iteration and restores the node to an online state.

## Related topics

- [Compute node health check on page 2677](#)

## Example health check script

As mentioned, the health check can be a shell script, PERL, Python, C-executable, or anything which can be executed from the command line capable of setting STDOUT. The example below demonstrates a very simple health check:

```
#!/bin/sh
/bin/mount | grep global
if [ $? != "0" ]
then
echo "ERROR cannot locate filesystem global"
fi
```

## Related topics

- [Compute node health check on page 2677](#)

## Debugging

TORQUE supports a number of diagnostic and debug options including the following:

*PBSDEBUG* environment variable - If set to 'yes', this variable will prevent *pbs\_server*, *pbs\_mom*, and/or *pbs\_sched* from backgrounding themselves allowing direct launch under a debugger. Also, some client commands will provide additional diagnostic information when this value is set.

*PBSLOGLEVEL* environment variable - Can be set to any value between 0 and 7 and specifies the logging verbosity level (default = 0)

*PBSCOREDUMP* environment variable - If set, it will cause the offending resource manager daemon to create a core file if a *SIGSEGV*, *SIGILL*, *SIGFPE*, *SIGSYS*, or *SIGTRAP* signal is received. The core dump will be placed in the daemon's home directory (*\$PBSHOME/mom\_priv* for *pbs\_mom* and *\$PBSHOME/server\_priv* for *pbs\_server*).

**i** To enable core dumping in a Red Hat system, you must add the following line to the `/etc/init.d/pbs_mom` and `/etc/init.d/pbs_server` scripts:

```
export DAEMON_COREFILE_LIMIT=unlimited
```

`NDEBUG #define` - if set at build time, will cause additional low-level logging information to be output to stdout for `pbs_server` and `pbs_mom` daemons.

`tracejob` reporting tool - can be used to collect and report logging and accounting information for specific jobs (for more information, see [Using "tracejob" to locate job failures on page 2668](#)).

**i** `PBSLOGLEVEL` and `PBSCOREDUMP` must be added to the `$PBSHOME/pbs_environment` file, not just the current environment. To set these variables, add a line to the `pbs_environment` file as either "variable=value" or just "variable". In the case of "variable=value", the environment variable is set up as the value specified. In the case of "variable", the environment variable is set based upon its value in the current environment.

## TORQUE error codes

Error code name	Number	Description
<b>PBSE_FLOOR</b>	15000	No error
<b>PBSE_UNKJOBID</b>	15001	Unknown job identifier
<b>PBSE_NOATTR</b>	15002	Undefined attribute
<b>PBSE_ATTRRO</b>	15003	Attempt to set READ ONLY attribute
<b>PBSE_IVALREQ</b>	15004	Invalid request
<b>PBSE_UNKREQ</b>	15005	Unknown batch request
<b>PBSE_TOOMANY</b>	15006	Too many submit retries
<b>PBSE_PERM</b>	15007	No permission
<b>PBSE_IFF_NOT_FOUND</b>	15008	"pbs_iff" not found; unable to authenticate
<b>PBSE_MUNGE_NOT_FOUND</b>	15009	"munge" executable not found; unable to authenticate
<b>PBSE_BADHOST</b>	15010	Access from host not allowed

Error code name	Number	Description
<b>PBSE_JOBEXIST</b>	15011	Job already exists
<b>PBSE_SYSTEM</b>	15012	System error occurred
<b>PBSE_INTERNAL</b>	15013	Internal server error occurred
<b>PBSE_REGROUTE</b>	15014	Parent job of dependent in rte queue
<b>PBSE_UNKSIG</b>	15015	Unknown signal name
<b>PBSE_BADATVAL</b>	15016	Bad attribute value
<b>PBSE_MODATRRUN</b>	15017	Cannot modify attribute in run state
<b>PBSE_BADSTATE</b>	15018	Request invalid for job state
<b>PBSE_UNKQUE</b>	15020	Unknown queue name
<b>PBSE_BADCRED</b>	15021	Invalid credential in request
<b>PBSE_EXPIRED</b>	15022	Expired credential in request
<b>PBSE_QUNOENB</b>	15023	Queue not enabled
<b>PBSE_QACCESS</b>	15024	No access permission for queue
<b>PBSE_BADUSER</b>	15025	Bad user - no password entry
<b>PBSE_HOPCOUNT</b>	15026	Max hop count exceeded
<b>PBSE_QUEEXIST</b>	15027	Queue already exists
<b>PBSE_ATTRRYPE</b>	15028	Incompatible queue attribute type
<b>PBSE_QUEBUSY</b>	15029	Queue busy (not empty)
<b>PBSE_QUENBIG</b>	15030	Queue name too long
<b>PBSE_NOSUP</b>	15031	Feature/function not supported

Error code name	Number	Description
<b>PBSE_QUENOEN</b>	15032	Cannot enable queue,needs add def
<b>PBSE_PROTOCOL</b>	15033	Protocol (ASN.1) error
<b>PBSE_BADATLST</b>	15034	Bad attribute list structure
<b>PBSE_NOCONNECTS</b>	15035	No free connections
<b>PBSE_NOSERVER</b>	15036	No server to connect to
<b>PBSE_UNKRESC</b>	15037	Unknown resource
<b>PBSE_EXCQRESC</b>	15038	Job exceeds queue resource limits
<b>PBSE_QUENODFLT</b>	15039	No default queue defined
<b>PBSE_NORERUN</b>	15040	Job not rerunnable
<b>PBSE_ROUTEREJ</b>	15041	Route rejected by all destinations
<b>PBSE_ROUTEEXPD</b>	15042	Time in route queue expired
<b>PBSE_MOMREJECT</b>	15043	Request to MOM failed
<b>PBSE_BADSCRIPT</b>	15044	(qsub) Cannot access script file
<b>PBSE_STAGEIN</b>	15045	Stage-In of files failed
<b>PBSE_RESCUNAV</b>	15046	Resources temporarily unavailable
<b>PBSE_BADGRP</b>	15047	Bad group specified
<b>PBSE_MAXQUED</b>	15048	Max number of jobs in queue
<b>PBSE_CKPBYSY</b>	15049	Checkpoint busy, may be retries
<b>PBSE_EXLIMIT</b>	15050	Limit exceeds allowable
<b>PBSE_BADACCT</b>	15051	Bad account attribute value

Error code name	Number	Description
<b>PBSE_ALRDYEXIT</b>	15052	Job already in exit state
<b>PBSE_NOCOPYFILE</b>	15053	Job files not copied
<b>PBSE_CLEANEOUT</b>	15054	Unknown job id after clean init
<b>PBSE_NOSYNCMSTR</b>	15055	No master in sync set
<b>PBSE_BADDEPEND</b>	15056	Invalid dependency
<b>PBSE_DUPLIST</b>	15057	Duplicate entry in list
<b>PBSE_DISPROTO</b>	15058	Bad DIS based request protocol
<b>PBSE_EXECTHERE</b>	15059	Cannot execute there
<b>PBSE_SISREJECT</b>	15060	Sister rejected
<b>PBSE_SISCOMM</b>	15061	Sister could not communicate
<b>PBSE_SVRDOWN</b>	15062	Requirement rejected -server shutting down
<b>PBSE_CKPSHORT</b>	15063	Not all tasks could checkpoint
<b>PBSE_UNKNODE</b>	15064	Named node is not in the list
<b>PBSE_UNKNODEATR</b>	15065	Node-attribute not recognized
<b>PBSE_NONODES</b>	15066	Server has no node list
<b>PBSE_NODENBIG</b>	15067	Node name is too big
<b>PBSE_NODEEXIST</b>	15068	Node name already exists
<b>PBSE_BADNDATVAL</b>	15069	Bad node-attribute value
<b>PBSE_MUTUALEX</b>	15070	State values are mutually exclusive
<b>PBSE_GMODERR</b>	15071	Error(s) during global modification of nodes

Error code name	Number	Description
<b>PBSE_NORELYMOM</b>	15072	Could not contact MOM
<b>PBSE_NOTSNODE</b>	15073	No time-shared nodes
<b>PBSE_JOBTYPE</b>	15074	Wrong job type
<b>PBSE_BADACLHOST</b>	15075	Bad ACL entry in host list
<b>PBSE_MAXUSERQUED</b>	15076	Maximum number of jobs already in queue for user
<b>PBSE_BADDISALLOWTYPE</b>	15077	Bad type in "disallowed_types" list
<b>PBSE_NOINTERACTIVE</b>	15078	Interactive jobs not allowed in queue
<b>PBSE_NOBATCH</b>	15079	Batch jobs not allowed in queue
<b>PBSE_NORERUNABLE</b>	15080	Rerunable jobs not allowed in queue
<b>PBSE_NONONRERUNABLE</b>	15081	Non-rerunable jobs not allowed in queue
<b>PBSE_UNKARRAYID</b>	15082	Unknown array ID
<b>PBSE_BAD_ARRAY_REQ</b>	15083	Bad job array request
<b>PBSE_TIMEOUT</b>	15084	Time out
<b>PBSE_JOBNOTFOUND</b>	15085	Job not found
<b>PBSE_NOFAULTTOLERANT</b>	15086	Fault tolerant jobs not allowed in queue
<b>PBSE_NOFAULTINTOLERANT</b>	15087	Only fault tolerant jobs allowed in queue
<b>PBSE_NOJOBARRAYS</b>	15088	Job arrays not allowed in queue
<b>PBSE_RELAYED_TO_MOM</b>	15089	Request was relayed to a MOM
<b>PBSE_MEM_MALLOC</b>	15090	Failed to allocate memory for memmgr
<b>PBSE_MUTEX</b>	15091	Failed to allocate controlling mutex (lock/unlock)

Error code name	Number	Description
<b>PBSE_TRHEADATTR</b>	15092	Failed to set thread attributes
<b>PBSE_THREAD</b>	15093	Failed to create thread
<b>PBSE_SELECT</b>	15094	Failed to select socket
<b>PBSE_SOCKET_FAULT</b>	15095	Failed to get connection to socket
<b>PBSE_SOCKET_WRITE</b>	15096	Failed to write data to socket
<b>PBSE_SOCKET_READ</b>	15097	Failed to read data from socket
<b>PBSE_SOCKET_CLOSE</b>	15098	Socket closed
<b>PBSE_SOCKET_LISTEN</b>	15099	Failed to listen in on socket
<b>PBSE_AUTH_INVALID</b>	15100	Invalid auth type in request
<b>PBSE_NOT_IMPLEMENTED</b>	15101	Functionality not yet implemented
<b>PBSE_QUEENOTAVAILABLE</b>	15102	Queue is not available

#### Related topics

- [Troubleshooting on page 2665](#)

## Appendices

The appendices provide tables of commands, parameters, configuration options, error codes, the Quick Start Guide, and so forth.

- [Appendix A: Commands overview on page 2686](#)
- [Appendix B: Server parameters on page 2766](#)
- [Appendix C: Node manager \(MOM\) configuration on page 2783](#)
- [Appendix D: Diagnostics and error codes on page 2802](#)
- [Appendix E: Considerations before upgrading on page 2810](#)
- [Appendix F: Large cluster considerations on page 2811](#)

- [Appendix G: Prologue and epilogue scripts on page 2817](#)
- [Appendix H: Running multiple TORQUE servers and MOMs on the same node on page 2825](#)
- [Appendix I: Security overview on page 2826](#)
- [Appendix J: Job submission filter \("qsub wrapper"\) on page 2827](#)
- [Appendix K: "torque.cfg" configuration file on page 2828](#)
- [Appendix L: TORQUE Quick Start Guide on page 2833](#)
- [Appendix M: BLCR acceptance tests on page 2836](#)

## Appendix A: Commands overview

### Client commands

Command	Description
<a href="#"><u>momctl</u></a>	Manage/diagnose MOM (node execution) daemon
<a href="#"><u>pbsdsh</u></a>	Launch tasks within a parallel job
<a href="#"><u>pbsnodes</u></a>	View/modify batch status of compute nodes
<a href="#"><u>qalter</u></a>	Modify queued batch jobs
<a href="#"><u>qchkpt</u></a>	Checkpoint batch jobs
<a href="#"><u>qdel</u></a>	Delete/cancel batch jobs
<a href="#"><u>qgpumode</u></a>	Specifies new mode for GPU
<a href="#"><u>qgpureset</u></a>	Reset the GPU
<a href="#"><u>qhold</u></a>	Hold batch jobs
<a href="#"><u>qmgr</u></a>	Manage policies and other batch configuration
<a href="#"><u>qmove on page 2729</u></a>	Move batch jobs

Command	Description
<a href="#">qorder</a> on page 2730	Exchange order of two batch jobs in any queue
<a href="#">qrerun</a>	Rerun a batch job
<a href="#">qrls</a>	Release batch job holds
<a href="#">qrun</a>	Start a batch job
<a href="#">qsig</a>	Send a signal to a batch job
<a href="#">qstat</a>	View queues and jobs
<a href="#">qsub</a>	Submit jobs
<a href="#">qterm</a>	Shutdown pbs server daemon
<b>tracejob</b>	Trace job actions and states recorded in TORQUE logs (see <a href="#">Using "tracejob" to locate job failures</a> on page 2668)

## Binary executables

Command	Description
<b>pbs_iff</b>	Interprocess authentication service
<a href="#">pbs_mom</a>	Start MOM (node execution) daemon
<a href="#">pbs_server</a>	Start server daemon
<a href="#">pbs_track</a>	Tell pbs_mom to track a new process

### Related topics

- [Appendix C: Node manager \(MOM\) configuration](#) on page 2783
- [Appendix B: Server parameters](#) on page 2766

## momctl

(PBS MOM Control)

## Synopsis

```

momctl -c { <JOBID> | all }
momctl -C
momctl -d { <INTEGER> | <JOBID> }
momctl -f <FILE>
momctl -h <HOST>[,<HOST>]...
momctl -p <PORT NUMBER>
momctl -q <ATTRIBUTE>
momctl -r { <FILE> | LOCAL:<FILE> }
momctl -s

```

## Overview

The `momctl` command allows remote shutdown, reconfiguration, diagnostics, and querying of the `pbs_mom` daemon.

## Format

<b>-c — Clear</b>	
<b>Format</b>	{ <JOBID>   <i>all</i> }
<b>Default</b>	---
<b>Description</b>	Makes the MOM unaware of the job's existence. It does not clean up any processes associated with the job.
<b>Example</b>	<code>momctl - node1 -c 15406</code>

<b>-C — Cycle</b>	
<b>Format</b>	---
<b>Default</b>	---
<b>Description</b>	Cycle <code>pbs_mom(s)</code>
<b>Example</b>	<code>momctl - node1 -C</code> Cycle <code>pbs_mom</code> on node1.

<b>-d — Diagnose</b>	
<b>Format</b>	{ <INTEGER>   <JOBID> }
<b>Default</b>	0
<b>Description</b>	Diagnose MOM(s) (For more details, see <a href="#">Diagnose detail on page 2692</a> below.)
<b>Example</b>	<pre>momctl - node1 -d 2</pre> Print level 2 and lower diagnose information for the MOM on node1.

<b>-f — Host File</b>	
<b>Format</b>	<FILE>
<b>Default</b>	---
<b>Description</b>	A file containing only comma or whitespace (space, tab, or new line) delimited hostnames
<b>Example</b>	<pre>momctl -f hosts.txt -d</pre> Print diagnose information for the MOMs running on the hosts specified in <code>hosts.txt</code> .

<b>-h — Host List</b>	
<b>Format</b>	<HOST>[,<HOST>]..
<b>Default</b>	localhost
<b>Description</b>	A comma separated list of hosts
<b>Example</b>	<pre>momctl -h node1,node2,node3 -d</pre> Print diagnose information for the MOMs running on node1, node2, and node3.

<b>-p — Port</b>	
<b>Format</b>	<PORT_NUMBER>

**-p — Port**

<b>Default</b>	TORQUE's default port number
<b>Description</b>	The port number for the specified MOM(s)
<b>Example</b>	<pre>momctl -p 5455 -h node1 -d</pre> <p>Request diagnose information over port 5455 on node1.</p>

**-q — Query**

<b>Format</b>	<ATTRIBUTE>
<b>Default</b>	---
<b>Description</b>	Query <ATTRIBUTE> on specified MOM, where <ATTRIBUTE> is a property listed by <a href="#">pbsnodes -a</a> (see <a href="#">Query attributes on page 2691</a> for a list of attributes)
<b>Example</b>	<pre>momctl -q phymem</pre> <p>Print the amount of phymem on localhost.</p>

**-r — Reconfigure**

<b>Format</b>	{ <FILE>   LOCAL:<FILE> }
<b>Default</b>	---
<b>Description</b>	Reconfigure MOM(s) with remote or local config file, <FILE>. This does not work if \$remote_reconf is not set to true when the MOM is started.
<b>Example</b>	<pre>momctl -r /home/user1/new.config -h node1</pre> <p>Reconfigure MOM on node1 with /home/user1/new.cofig on node1.</p>

**-s — Shutdown**

<b>Format</b>	
<b>Default</b>	---

<b>-s — Shutdown</b>	
<b>Description</b>	Shutdown pbs_mom
<b>Example</b>	<pre>momctl -s</pre> Terminates pbs_mom process on localhost.

## Query attributes

<b>Attribute</b>	<b>Description</b>
<b>arch</b>	node hardware architecture
<b>availmem</b>	available RAM
<b>loadave</b>	1 minute load average
<b>ncpus</b>	number of CPUs available on the system
<b>netload</b>	total number of bytes transferred over all network interfaces
<b>nsessions</b>	number of sessions active
<b>nusers</b>	number of users active
<b>physmem</b>	configured RAM
<b>sessions</b>	list of active sessions
<b>totmem</b>	configured RAM plus configured swap

## Diagnose detail

Level	Description
0	<p>Display the following information:</p> <ul style="list-style-type: none"> <li>• Local hostname</li> <li>• Expected server hostname</li> <li>• Execution version</li> <li>• MOM home directory</li> <li>• MOM config file version (if specified)</li> <li>• Duration MOM has been executing</li> <li>• Duration since last request from pbs_server daemon</li> <li>• Duration since last request to pbs_server daemon</li> <li>• RM failure messages (if any)</li> <li>• Log verbosity level</li> <li>• Local job list</li> </ul>
1	<p>All information for level 0 plus the following:</p> <ul style="list-style-type: none"> <li>• Interval between updates sent to server</li> <li>• Number of initialization messages sent to pbs_server daemon</li> <li>• Number of initialization messages received from pbs_server daemon</li> <li>• Prolog/epilog alarm time</li> <li>• List of trusted clients</li> </ul>
2	<p>All information from level 1 plus the following:</p> <ul style="list-style-type: none"> <li>• PID</li> <li>• Event alarm status</li> </ul>
3	<p>All information from level 2 plus the following:</p> <ul style="list-style-type: none"> <li>• syslog enabled</li> </ul>

**Example 4-27: MOM diagnostics**

```

momctl -d 1

Host: nsrc/nsrc.fllcl.com      Server: 10.10.10.113      Version: torque_1.1.0p4
HomeDirectory:                /usr/spool/PBS/mom_priv
ConfigVersion:                147
MOM active:                   7390 seconds
Last Msg From Server:        7389 seconds (CLUSTER_ADDRS)
Server Update Interval:      20 seconds
Server Update Interval:      20 seconds
Init Msgs Received:          0 hellos/1 cluster-addr
Init Msgs Sent:              1 hellos
LOGLEVEL:                    0 (use SIGUSR1/SIGUSR2 to adjust)
Prolog Alarm Time:           300 seconds
Trusted Client List:         12.14.213.113,127.0.0.1
JobList:                     NONE

diagnostics complete

```

**Example 4-28: System shutdown**

```

> momctl -s -f /opt/clusterhostfile

shutdown request successful on node001
shutdown request successful on node002
shutdown request successful on node003
shutdown request successful on node004
shutdown request successful on node005
shutdown request successful on node006

```

## pbs\_mom

Start a pbs batch execution mini-server.

### Synopsis

```

pbs_mom [-a alarm] [-A alias] [-C chkdiritory] [-c config] [-d directory] [-h
hostname]
[-L logfile] [-M MOMport] [-R RPPport] [-p|-r] [-P purge] [-w] [-x]

```

### Description

The `pbs_mom` command is located within the `TORQUE_HOME` directory and starts the operation of a batch Machine Oriented Mini-server (MOM) on the execution host. To ensure that the `pbs_mom` command is not runnable by the general user community, the server will only execute if its real and effective uid is zero.

The first function of `pbs_mom` is to place jobs into execution as directed by the server, establish resource usage limits, monitor the job's usage, and notify the server when the job completes. If they exist, `pbs_mom` will execute a prologue script before executing a job and an epilogue script after executing the job.

The second function of `pbs_mom` is to respond to resource monitor requests. This was done by a separate process in previous versions of PBS but has now been combined into one process. It provides information about the status of running jobs, memory available, etc.

The last function of `pbs_mom` is to respond to task manager requests. This involves communicating with running tasks over a TCP socket as well as communicating with other MOMs within a job (a.k.a. a "sisterhood").

`pbs_mom` will record a diagnostic message in a log file for any error occurrence. The log files are maintained in the `mom_logs` directory below the home directory of the server. If the log file cannot be opened, the diagnostic message is written to the system console.

## Options

Flag	Name	Description
<b>-a</b>	alarm	Used to specify the alarm timeout in seconds for computing a resource. Every time a resource request is processed, an alarm is set for the given amount of time. If the request has not completed before the given time, an alarm signal is generated. The default is 5 seconds.
<b>-A</b>	alias	Used to specify this multimom's alias name. The alias name needs to be the same name used in the <code>mom.hierarchy</code> file. It is only needed when running multiple MOMs on the same machine. For more information, see <a href="#">TORQUE Multi-MOM on page 2580</a> .
<b>-C</b>	chkdirectory	Specifies The path of the directory used to hold checkpoint files. (Currently this is only valid on Cray systems.) The default directory is <code>TORQUE_HOME/spool/checkpoint</code> (see the <b>-d</b> option). The directory specified with the <b>-C</b> option must be owned by root and accessible (rwx) only by root to protect the security of the checkpoint files.
<b>-c</b>	config	Specifies an alternative configuration file, see description below. If this is a relative file name it will be relative to <code>TORQUE_HOME/mom_priv</code> , (see the <b>-d</b> option). If the specified file cannot be opened, <code>pbs_mom</code> will abort. If the <b>-C</b> option is not supplied, <code>pbs_mom</code> will attempt to open the default configuration file "config" in <code>TORQUE_HOME/mom_priv</code> . If this file is not present, <code>pbs_mom</code> will log the fact and continue.
<b>-d</b>	directory	Specifies the path of the directory which is the home of the server's working files, <code>TORQUE_HOME</code> . This option is typically used along with <b>-M</b> when debugging MOM. The default directory is given by <code>\$PBS_SERVER_HOME</code> which is typically <code>/usr/spool/PBS</code> .
<b>-h</b>	hostname	Set MOM's hostname. This can be useful on multi-homed networks.
<b>-L</b>	logfile	Specify an absolute path name for use as the log file. If not specified, MOM will open a file named for the current date in the <code>TORQUE_HOME/mom_logs</code> directory (see the <b>-d</b> option).
<b>-M</b>	port	Specifies the port number on which the mini-server (MOM) will listen for batch requests.

Flag	Name	Description
<b>-p</b>	n/a	Specifies the impact on jobs which were in execution when the mini-server shut down. On any restart of MOM, the new mini-server will not be the parent of any running jobs, MOM has lost control of her offspring (not a new situation for a mother). With the <b>-p</b> option, MOM will allow the jobs to continue to run and monitor them indirectly via polling. This flag is redundant in that this is the default behavior when starting the server. The <b>-p</b> option is mutually exclusive with the <b>-R</b> and <b>-q</b> options.
<b>-P</b>	purge	Specifies the impact on jobs which were in execution when the mini-server shut down. With the <b>-P</b> option, it is assumed that either the entire system has been restarted or the MOM has been down so long that it can no longer guarantee that the pid of any running process is the same as the recorded job process pid of a recovering job. Unlike the <b>-p</b> option, no attempt is made to try and preserve or recover running jobs. All jobs are terminated and removed from the queue.
<b>-q</b>	n/a	Specifies the impact on jobs which were in execution when the mini-server shut down. With the <b>-q</b> option, MOM will allow the processes belonging to jobs to continue to run, but will not attempt to monitor them. The <b>-q</b> option is mutually exclusive with the <b>-p</b> and <b>-R</b> options.
<b>-R</b>	port	Specifies the port number on which the mini-server (MOM) will listen for resource monitor requests, task manager requests and inter-MOM messages. Both a UDP and a TCP port of this number will be used.
<b>-r</b>	n/a	Specifies the impact on jobs which were in execution when the mini-server shut down. With the <b>-r</b> option, MOM will kill any processes belonging to jobs, mark the jobs as terminated, and notify the batch server which owns the job. The <b>-r</b> option is mutually exclusive with the <b>-p</b> and <b>-q</b> options.  Normally the mini-server is started from the system boot file without the <b>-p</b> or the <b>-r</b> option. The mini-server will make no attempt to signal the former session of any job which may have been running when the mini-server terminated. It is assumed that on reboot, all processes have been killed. If the <b>-r</b> option is used following a reboot, process IDs (pids) may be reused and MOM may kill a process that is not a batch session.
<b>-w</b>	wait_for_server	When started with <b>-w</b> , pbs_moms wait until they get their MOM hierarchy file from <a href="#">pbs_server</a> to send their first update, or until 10 minutes pass. This reduces network traffic on startup and can bring up clusters faster.
<b>-x</b>	n/a	Disables the check for privileged port resource monitor connections. This is used mainly for testing since the privileged port is the only mechanism used to prevent any ordinary user from connecting.

## Configuration file

The configuration file, located at `mom_priv/config` by default, can be specified on the command line at program start with the `-C` flag. The use of this file is to provide several types of run time information to `pbs_mom`: static resource names and values, external resources provided by a program to be run on request via a shell escape, and values to pass to internal set up functions at initialization (and re-initialization).

See the [Parameters on page 2784](#) page for a full list of `pbs_mom` parameters.

Each item type is on a single line with the component parts separated by white space. If the line starts with a hash mark (pound sign, #), the line is considered to be a comment and is skipped.

### *Static Resources*

For static resource names and values, the configuration file contains a list of resource names/values pairs, one pair per line and separated by white space. An example of static resource names and values could be the number of tape drives of different types and could be specified by:

- tape3480 4
- tape3420 2
- tapedat 1
- tape8mm 1

### *Shell Commands*

If the first character of the value is an exclamation mark (!), the entire rest of the line is saved to be executed through the services of the `system(3)` standard library routine.

The shell escape provides a means for the resource monitor to yield arbitrary information to the scheduler. Parameter substitution is done such that the value of any qualifier sent with the query, as explained below, replaces a token with a percent sign (%) followed by the name of the qualifier. For example, here is a configuration file line which gives a resource name of "escape":

```
escape !echo %xxx %yyy
```

If a query for "escape" is sent with no qualifiers, the command executed would be `echo %xxx %yyy`.

If one qualifier is sent, `escape[xxx=hi there]`, the command executed would be `echo hi there %yyy`.

If two qualifiers are sent, `escape[xxx=hi][yyy=there]`, the command executed would be `echo hi there`.

If a qualifier is sent with no matching token in the command line, `escape[zzz=snafu]`, an error is reported.

## Resources

Resource Manager queries can be made with `momctl -q` options to retrieve and set `pbs_mom` options. Any configured static resource may be retrieved with a request of the same name. These are resource requests not otherwise documented in the PBS ERS.

Request	Description
<b>cycle</b>	Forces an immediate MOM cycle.
<b>status_update_time</b>	Retrieve or set the \$status_update_time parameter.
<b>check_poll_time</b>	Retrieve or set the \$check_poll_time parameter.
<b>configversion</b>	Retrieve the config version.
<b>jobstartblocktime</b>	Retrieve or set the \$jobstartblocktime parameter.
<b>enablemomrestart</b>	Retrieve or set the \$enablemomrestart parameter.
<b>loglevel</b>	Retrieve or set the \$loglevel parameter.
<b>down_on_error</b>	Retrieve or set the EXPERIMENTAL \$down_on_error parameter.
<b>diag0 - diag4</b>	Retrieves varied diagnostic information.
<b>rcpcmd</b>	Retrieve or set the \$rcpcmd parameter.
<b>version</b>	Retrieves the pbs_mom version.

## Health check

The health check script is executed directly by the pbs\_mom daemon under the root user id. It must be accessible from the compute node and may be a script or compiled executable program. It may make any needed system calls and execute any combination of system utilities but should not execute resource manager client commands. Also, the pbs\_mom daemon blocks until the health check is completed and does not possess a built-in timeout. Consequently, it is advisable to keep the launch script execution time short and verify that the script will not block even under failure conditions.

If the script detects a failure, it should return the keyword "Error" to `stdout` followed by an error message. The message (up to 256 characters) immediately following the Error string will be assigned to the node attribute message of the associated node.

If the script detects a failure when run from "jobstart", then the job will be rejected. You can use this behavior with an advanced scheduler, such as Moab Workload Manager, to cause the job to be routed to another node. TORQUE currently ignores Error messages by default, but you can configure an advanced scheduler to react appropriately.

If the experimental \$down\_on\_error MOM setting is enabled, the MOM will set itself to state down and report to pbs\_server. Additionally, the experimental \$down\_on\_error server attribute can be enabled which has the same effect but moves the decision to pbs\_server. It is redundant to have MOM's

`$down_on_error` and `pbs_servers down_on_error` features enabled. See "down\_on\_error" in `pbs_server_attributes(7B)`.

## Files

File	Description
<code>\$PBS_SERVER_HOME/server_name</code>	Contains the hostname running <code>pbs_server</code>
<code>\$PBS_SERVER_HOME/mom_priv</code>	The default directory for configuration files, typically <code>(/usr/spool/pbs)/mom_priv</code>
<code>\$PBS_SERVER_HOME/mom_logs</code>	Directory for log files recorded by the server
<code>\$PBS_SERVER_HOME/mom_priv/-prologue</code>	The administrative script to be run before job execution
<code>\$PBS_SERVER_HOME/mom_priv/e-epilogue</code>	The administrative script to be run after job execution

## Signal handling

`pbs_mom` handles the following signals:

Signal	Description
<b>SIGHUP</b>	Causes <code>pbs_mom</code> to re-read its configuration file, close and reopen the log file, and reinitialize resource structures.
<b>SIGALRM</b>	Results in a log file entry. The signal is used to limit the time taken by certain children processes, such as the prologue and epilogue.
<b>SIGINT and SIGTERM</b>	Results in <code>pbs_mom</code> exiting without terminating any running jobs. This is the action for the following signals as well: <code>SIGXCPU</code> , <code>SIGXFSZ</code> , <code>SIGCPULIM</code> , and <code>SIGSHUTDN</code> .
<b>SIGUSR1, SIGUSR2</b>	Causes the MOM to increase and decrease logging levels, respectively.
<b>SIGPIPE, SIGINFO</b>	Are ignored.
<b>SIGBUS, SIGFPE, SIGILL, SIGTRAP, and SIGSYS</b>	Cause a core dump if the <code>PBSCOREDUMP</code> environmental variable is defined.

All other signals have their default behavior installed.

## Exit status

If the `pbs_mom` command fails to begin operation, the server exits with a value greater than zero.

### Related topics

- [pbs\\_server\(8B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_scheduler\\_basl\(8B\)](#)
- [pbs\\_scheduler\\_tcl\(8B\)](#)
- PBS External Reference Specification
- PBS Administrators Guide

## pbs\_server

(*PBS Server*) pbs batch system manager

### Synopsis

```
pbs_server [-a active] [-c] [-d config_path] [-f force overwrite] [-p port] [-A acctfile]
[-l location] [-L logfile] [-S scheduler_port]
[-h hostname] [-t type] [--ha]
[-n don't send hierarchy] [--about] [-v] [--version]
```

### Description

The `pbs_server` command starts the operation of a batch server on the local host. Typically, this command will be in a local boot file such as `/etc/rc.local`. If the batch server is already in execution, `pbs_server` will exit with an error. To ensure that the `pbs_server` command is not runnable by the general user community, the server will only execute if its real and effective uid is zero.

The server will record a diagnostic message in a log file for any error occurrence. The log files are maintained in the `server_logs` directory below the home directory of the server. If the log file cannot be opened, the diagnostic message is written to the system console.

As of TORQUE 4.0, the `pbs_server` is multi-threaded which leads to quicker response to client commands, is more robust, and allows for higher job throughput.

## Options

Option	Name	Description
<b>-A</b>	acctfile	Specifies an absolute path name of the file to use as the accounting file. If not specified, the file name will be the current date in the <code>PBS_HOME/server_priv/accounting</code> directory.
<b>-a</b>	active	Specifies if scheduling is active or not. This sets the server attribute <code>scheduling</code> . If the option argument is "true" ("True", "t", "T", or "1"), the server is active and the PBS job scheduler will be called. If the argument is "false" ("False", "f", "F", or "0"), the server is idle, and the scheduler will not be called and no jobs will be run. If this option is not specified, the server will retain the prior value of the scheduling attribute.
<b>-c</b>	wait_for_moms	This directs <code>pbs_server</code> to send the MOM hierarchy only to MOMs that request it for the first 10 minutes. After 10 minutes, it attempts to send the MOM hierarchy to MOMs that haven't requested it already. This greatly reduces traffic on start up.
<b>-d</b>	config_directory	Specifies the path of the directory which is home to the server's configuration files, <code>PBS_HOME</code> . A host may have multiple servers. Each server must have a different configuration directory. The default configuration directory is given by the symbol <code>\$PBS_SERVER_HOME</code> which is typically <code>var/spool/torque</code> .
<b>-f</b>	force overwrite	Forces an overwrite of the server database. This can be useful to bypass the <code>yes/no</code> prompt when running something like <code>pbs_server -t create</code> and can ease installation and configuration of TORQUE via scripts.
<b>-h</b>	hostname	Causes the server to start under a different hostname as obtained from <code>gethostname(2)</code> . Useful for servers with multiple network interfaces to support connections from clients over an interface that has a hostname assigned that differs from the one that is returned by <code>gethostname(2)</code> .
<b>--ha</b>	high_availability	Starts server in high availability mode (for details, see <a href="#">Server high availability on page 2640</a> ).
<b>-L</b>	logfile	Specifies an absolute path name of the file to use as the log file. If not specified, the file will be the current date in the <code>PBS_HOME/server_logs</code> directory (see the <a href="#">-d</a> option).
<b>-l</b>	location	Specifies where to find Moab when it does not reside on the same host as TORQUE.
<b>-n</b>	no send	This directs <code>pbs_server</code> to not send the hierarchy to all the MOMs on startup. Instead, the hierarchy is only sent if a MOM requests it. This flag works only in conjunction with the <a href="#">local MOM hierarchy</a> feature.

Option	Name	Description
<b>-p</b>	port	Specifies the port number on which the server will listen for batch requests. If multiple servers are running on a single host, each must have its own unique port number. This option is for use in testing with multiple batch systems on a single host.
<b>-S</b>	scheduler_ port	Specifies the port number to which the server should connect when contacting the scheduler. The argument scheduler_conn is of the same syntax as under the -M option.
<b>-t</b>	type	<p>Specifies the impact on jobs which were in execution, running, when the server shut down. If the running job is not rerunnable or restartable from a checkpoint image, the job is aborted. If the job is rerunnable or restartable, then the actions described below are taken. When the type argument is:</p> <ul style="list-style-type: none"> <li>• <i>hot</i> – All jobs are queued except non-rerunnable jobs that were executing. Any rerunnable job which was executing when the server went down will be run immediately. This returns the server to the same state as when it went down. After those jobs are restarted, then normal scheduling takes place for all remaining queued jobs.</li> </ul> <p>If a job cannot be restarted immediately because of a missing resource, such as a node being down, the server will attempt to restart it periodically for up to 5 minutes. After that period, the server will revert to a normal state, as if warm started, and will no longer attempt to restart any remaining jobs which were running prior to the shutdown.</p> <ul style="list-style-type: none"> <li>• <i>warm</i> – All rerunnable jobs which were running when the server went down are queued. All other jobs are maintained. New selections are made for which jobs are placed into execution. Warm is the default if <i>-t</i> is not specified.</li> <li>• <i>cold</i> – All jobs are deleted. Positive confirmation is required before this direction is accepted.</li> <li>• <i>create</i> – The server will discard any existing configuration files, queues and jobs, and initialize configuration files to the default values. The server is idled.</li> </ul>

## Files

File	Description
<b>TORQUE_HOME/server_ priv</b>	Default directory for configuration files, typically /usr/spool/pbs/server_ priv
<b>TORQUE_HOME/server_ logs</b>	Directory for log files recorded by the server

## Signal handling

On receipt of the following signals, the server performs the defined action:

Action	Description
<b>SIGHUP</b>	The current server log and accounting log are closed and reopened. This allows for the prior log to be renamed and a new log started from the time of the signal.
<b>SIGINT</b>	Causes an orderly shutdown of pbs_server.
<b>SIGUSR1, SIGUSR2</b>	Causes server to increase and decrease logging levels, respectively.
<b>SIGTERM</b>	Causes an orderly shutdown of pbs_server.
<b>SIGSHUTDN</b>	On systems (Unicos) where SIGSHUTDN is defined, it also causes an orderly shutdown of the server.
<b>SIGPIPE</b>	This signal is ignored.

All other signals have their default behavior installed.

## Exit status

If the server command fails to begin batch operation, the server exits with a value greater than zero.

### Related topics

- [pbs\\_mom](#)(8B)
- [pbsnodes](#)(8B)
- [qmgr](#)(1B)
- [qrun](#)(8B)
- [qsub](#)(1B)
- [qterm](#)(8B)

### Non-Adaptive Computing topics

- [pbs\\_connect](#)(3B)
- [pbs\\_sched\\_basl](#)(8B)
- [pbs\\_sched\\_tcl](#)(8B)
- [qdisable](#)(8B)
- [qenable](#)(8B)
- [qstart](#)(8B)
- [qstop](#)(8B)
- PBS External Reference Specification

## pbs\_track

Starts a new process and informs pbs\_mom to start tracking it.

## Synopsis

```
pbs_track -j <JOBID> [-b] <executable> [args]
```

## Description

The `pbs_track` command tells a `pbs_mom` daemon to monitor the lifecycle and resource usage of the process that it launches using `exec()`. The `pbs_mom` is told about this new process via the Task Manager API, using `tm_adopt()`. The process must also be associated with a job that already exists on the `pbs_mom`.

By default, `pbs_track` will send its PID to TORQUE via `tm_adopt()`. It will then perform an `exec()`, causing `<executable>` to run with the supplied arguments. `pbs_track` will not return until the launched process has completed because it becomes the launched process.

This command can be considered related to the [pbsdsh](#) command which uses the `tm_spawn()` API call. The `pbsdsh` command asks a `pbs_mom` to launch and track a new process on behalf of a job. When it is not desirable or possible for the `pbs_mom` to spawn processes for a job, `pbs_track` can be used to allow an external entity to launch a process and include it as part of a job.

This command improves integration with TORQUE and SGI's MPT MPI implementation.

## Options

Option	Description
<b>-j</b> <b>&lt;JOBID&gt;</b>	Job ID the new process should be associated with.
<b>-b</b>	Instead of having <code>pbs_track</code> send its PID to TORQUE, it will <code>fork()</code> first, send the child PID to TORQUE, and then execute from the forked child. This essentially "backgrounds" <code>pbs_track</code> so that it will return after the new process is launched.

## Operands

The `pbs_track` command accepts a path to a program/executable (`<executable>`) and, optionally, one or more arguments to pass to that program.

## Exit status

Because the `pbs_track` command becomes a new process (if used without **-b**), its exit status will match that of the new process. If the **-b** option is used, the exit status will be zero if no errors occurred before launching the new process.

If `pbs_track` fails, whether due to a bad argument or other error, the exit status will be set to a non-zero value.

## Related topics

- [pbsdsh\(1B\)](#)

## Non-Adaptive Computing topics

- [tm\\_spawn\(3B\)](#)

## pbsdsh

The `pbsdsh` command distributes tasks to nodes under `pbs`.

## Synopsis

```
pbsdsh [-c copies] [-o] [-s] [-u] [-v] program [args]
pbsdsh [-n node] [-o] [-s] [-u] [-v] program [args]
pbsdsh [-h nodename] [-o] [-v] program [args]
```

## Description

Executes (spawns) a normal Unix program on one or more nodes under control of the Portable Batch System, PBS. Pbsdsh uses the Task Manager API (see `tm_spawn(3)`) to distribute the program on the allocated nodes.

When run without the `-c` or the `-n` option, `pbsdsh` will spawn the program on all nodes allocated to the PBS job. The spawns take place concurrently – all execute at (about) the same time.

Users will find the `PBS_TASKNUM`, `PBS_NODENUM`, and the `PBS_VNODENUM` environmental variables useful. They contain the TM task id, the node identifier, and the `cpu` (virtual node) identifier.

**i** Note that under particularly high workloads, the `pbsdsh` command may not function properly.

## Options

Option	Name	Description
<code>-c</code>	<code>copies</code>	The program is spawned on the first <code>Copies</code> nodes allocated. This option is mutually exclusive with <code>-n</code> .
<code>-h</code>	<code>hostname</code>	The program is spawned on the node specified.
<code>-n</code>	<code>node</code>	The program is spawned on one node which is the <code>n</code> -th node allocated. This option is mutually exclusive with <code>-c</code> .
<code>-o</code>	<code>---</code>	Capture stdout of the spawned program. Normally stdout goes to the job's output.

Option	Name	Description
<b>-s</b>	---	If this option is given, the program is run in turn on each node, one after the other.
<b>-u</b>	---	The program is run once on each node (unique). This ignores the number of allocated processors on a given node.
<b>-v</b>	---	Verbose output about error conditions and task exit status is produced.

## Operands

The first operand, `program`, is the program to execute.

Additional operands are passed as arguments to the program.

## Standard error

The `pbsdsh` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the command, the exit status will be a value of zero.

If the `pbsdsh` command fails to process any operand, or fails to contact the MOM daemon on the localhost the command exits with a value greater than zero.

Related topics

- [qsub\(1B\)](#)

## Non-Adaptive Computing topics

- [tm\\_spawn\(3B\)](#)

## pbsnodes

PBS node manipulation.

## Synopsis

```
pbsnodes [-{a|x}] [-q] [-s server] [node:property]
pbsnodes -l [-q] [-s server] [state] [nodename:property ...]
pbsnodes -m <running|standby|suspend|hibernate|shutdown> <host list>
pbsnodes [-{c|d|o|r}] [-q] [-s server] [-n -l] [-N "note"] [node:property]
```

## Description

The `pbsnodes` command is used to mark nodes down, free or offline. It can also be used to list nodes and their state. Node information is obtained by sending a request to the PBS job server. Sets of nodes can be operated on at once by specifying a node property prefixed by a colon. (For more information, see Node states.)

Nodes do not exist in a single state, but actually have a set of states. For example, a node can be simultaneously "busy" and "offline". The "free" state is the absence of all other states and so is never combined with other states.

In order to execute `pbsnodes` with other than the `-a` or `-l` options, the user must have PBS Manager or Operator privilege.

## Options

	Description
-a	All attributes of a node or all nodes are listed. This is the default if no flag is given.
-x	Same as <code>-a</code> , but the output has an XML-like format.
-c	Clear OFFLINE from listed nodes.
-C	Print MOM diagnosis on the listed nodes. Not yet implemented. Use <code>momctl</code> instead.

## Description

- Set the hosts in the specified host list to the requested power state. If a compute node does not support the energy-saving power state you request, the command returns an error and leaves the state unchanged. In order for the command to wake a node from a low-power state, Wake-on-LAN (WOL) must be enabled for the node.

**i** In order for the command to wake a node from a low-power state, Wake-on-LAN must be enabled for the node and it must support the `g` WOL packet. For more information, see [Changing node power states](#) on page 2619.

The allowable power states are:

- **Running:** The node is up and running.
- **Standby:** CPU is halted but still powered. Moderate power savings but low latency entering and leaving this state.
- **Suspend:** Also known as Suspend-to-RAM. Machine state is saved to RAM. RAM is put into self-refresh mode. Much more significant power savings with longer latency entering and leaving state.
- **Hibernate:** Also known as Suspend-to-disk. Machine state is saved to disk and then powered down. Significant power savings but very long latency entering and leaving state.
- **Shutdown:** Equivalent to `shutdown now` command as root.

The host list is a space-delimited list of node host names.

## Description

```
pbsnodes -m shutdown node01 node02 node03 node04
```

*With this command, pbs\_server tells the pbs\_mom associated with nodes01-04 to shut down the node.*

The `pbsnodes` output shows the current power state of nodes. In this example, note that `pbsnodes` returns the MAC addresses of the nodes.

```
pbsnodes
nuc1
  state = free
  power_state = Running
  np = 4
  ntype = cluster
  status = rectime=1395765676,macaddr=0b:25:22:92:7b:26
, cpuclock=Fixed, varattr=, jobs=, state=free, netload=1242652020, gres=, loadave=0.16, ncpus=6, physmem=
=16435852kb, availmem=24709056kb, totmem=33211016kb, idletime=4636, nusers=3, nsessions=12, sessions=
2758 998 1469 2708 2797 2845 2881 2946 4087 4154 4373 6385, uname=Linux bdaw 3.2.0-60-generic
#91-Ubuntu SMP Wed Feb 19 03:54:44 UTC 2014 x86_64, opsys=linux
  note = This is a node note
  mom_service_port = 15002
  mom_manager_port = 15003

nuc2
  state = free
  power_state = Running
  np = 4
  ntype = cluster
  status = rectime=1395765678,macaddr=2c:a8:6b:f4:b9:35
, cpuclock=OnDemand:800MHz, varattr=, jobs=, state=free, netload=12082362, gres=, loadave=0.00, ncpus=4
, physmem=16300576kb, availmem=17561808kb, totmem=17861144kb, idletime=67538, nusers=2, nsessions=7, s
essions=2189 2193 2194 2220 2222 2248 2351, uname=Linux nuc2 2.6.32-431.el6.x86_64 #1 SMP Fri
Nov 22 03:15:09 UTC 2013 x86_64, opsys=linux
  mom_service_port = 15002
  mom_manager_port = 15003
```

- Add the OFFLINE state. This is different from being marked DOWN. OFFLINE prevents new jobs from running on the specified nodes. This gives the administrator a tool to hold a node out of service without changing anything else. The OFFLINE state will never be set or cleared automatically by `pbs_server`; it is purely for the manager or operator.
- Purge the node record from `pbs_server`. Not yet implemented.
- Reset the listed nodes by clearing OFFLINE and adding DOWN state. `pbs_server` will ping the node and, if they communicate correctly, free the node.

	Description
I	<ul style="list-style-type: none"> <li>- List node names and their state. If no state is specified, only nodes in the DOWN, OFFLINE, or UNKNOWN states are listed. Specifying a state string acts as an output filter. Valid state strings are "active", "all", "busy", "down", "free", "job-exclusive", "job-sharing", "offline", "reserve", "state-unknown", "time-shared", and "up". <ul style="list-style-type: none"> <li>• Using <i>all</i> displays all nodes and their attributes.</li> <li>• Using <i>active</i> displays all nodes which are job-exclusive, job-sharing, or busy.</li> <li>• Using <i>up</i> displays all nodes in an "up state". Up states include job-exclusive, job-sharing, reserve, free, busy and time-shared.</li> <li>• All other strings display the nodes which are currently in the state indicated by the string.</li> </ul> </li> </ul>
M	<ul style="list-style-type: none"> <li>- Specify a "note" attribute. This allows an administrator to add an arbitrary annotation to the listed nodes. To clear a note, use <code>-N ""</code> or <code>-N n</code>.</li> </ul>
I	<ul style="list-style-type: none"> <li>- Show the "note" attribute for nodes that are DOWN, OFFLINE, or UNKNOWN. This option requires <a href="#">-l</a>.</li> </ul>
C	<ul style="list-style-type: none"> <li>- Suppress all error messages.</li> </ul>
s	<ul style="list-style-type: none"> <li>- Specify the PBS server's hostname or IP address.</li> </ul>

#### Related topics

- [pbs\\_server\(8B\)](#)

#### Non-Adaptive Computing topics

- PBS External Reference Specification

## qalter

Alter batch job.

### Synopsis

```
qalter [-a date_time] [-A account_string] [-c interval] [-e path_name]
[-h hold_list] [-j join_list] [-k keep_list] [-l resource_list]
[-m mail_options] [-M mail_list] [-n] [-N name] [-o path_name]
[-p priority] [-r y|n] [-S path_name_list] [-u user_list]
[-v variable_list] [-W additional_attributes]
```

```
[-t array_range]
job_identifier ...
```

## Description

The `qalter` command modifies the attributes of the job or jobs specified by `job_identifier` on the command line. Only those attributes listed as options on the command will be modified. If any of the specified attributes cannot be modified for a job for any reason, none of that job's attributes will be modified.

The `qalter` command accomplishes the modifications by sending a Modify Job batch request to the batch server which owns each job.

## Options

Option	Name	Description
<b>-a</b>	date_time	<p>Replaces the time at which the job becomes eligible for execution. The <code>date_time</code> argument syntax is:</p> <pre>[[ [ [CC] YY] MM] DD] hhmm [ .SS]</pre> <p>If the month, <code>MM</code>, is not specified, it will default to the current month if the specified day <code>DD</code>, is in the future. Otherwise, the month will be set to next month. Likewise, if the day, <code>DD</code>, is not specified, it will default to today if the time <code>hhmm</code> is in the future. Otherwise, the day will be set to tomorrow.</p> <p>This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.</p>
<b>-A</b>	account_string	<p>Replaces the account string associated with the job. This attribute cannot be altered once the job has begun execution.</p>

Option	Name	Description
-c	checkpoint_interval	<p>Replaces the interval at which the job will be checkpointed. If the job executes upon a host which does not support checkpointing, this option will be ignored.</p> <p>The interval argument is specified as:</p> <ul style="list-style-type: none"> <li>• <i>n</i> – No checkpointing is to be performed.</li> <li>• <i>s</i> – Checkpointing is to be performed only when the server executing the job is shutdown.</li> <li>• <i>c</i> – Checkpointing is to be performed at the default minimum cpu time for the queue from which the job is executing.</li> <li>• <i>c=minutes</i> – Checkpointing is performed at intervals of the specified amount of time in minutes. Minutes are the number of minutes of CPU time used, not necessarily clock time.</li> </ul> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 10px 0;"> <p> This value must be greater than zero. If the number is less than the default checkpoint time, the default time will be used.</p> </div> <p>This attribute can be altered once the job has begun execution, but the new value does not take effect unless the job is rerun.</p>
-e	path_name	<p>Replaces the path to be used for the standard error stream of the batch job. The path argument is of the form:</p> <p>[hostname:]path_name</p> <p>where <i>hostname</i> is the name of a host to which the file will be returned and <i>path_name</i> is the path name on that host in the syntax recognized by POSIX 1003.1. The argument will be interpreted as follows:</p> <ul style="list-style-type: none"> <li>• <i>path_name</i> – Where <i>path_name</i> is not an absolute path name, then the <code>qalter</code> command will expand the path name relative to the current working directory of the command. The command will supply the name of the host upon which it is executing for the <i>hostname</i> component.</li> <li>• <i>hostname:path_name</i> – Where <i>path_name</i> is not an absolute path name, then the <code>qalter</code> command will not expand the path name. The execution server will expand it relative to the home directory of the user on the system specified by <i>hostname</i>.</li> </ul> <p>This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.</p>

Option	Name	Description
<b>-h</b>	hold_list	<p>Updates the types of holds on the job. The hold_list argument is a string of one or more of the following characters:</p> <ul style="list-style-type: none"> <li>• <i>u</i> – Add the USER type hold.</li> <li>• <i>s</i> – Add the SYSTEM type hold if the user has the appropriate level of privilege. (Typically reserved to the batch administrator.)</li> <li>• <i>o</i> – Add the OTHER (or OPERATOR ) type hold if the user has the appropriate level of privilege. (Typically reserved to the batch administrator and batch operator.)</li> <li>• <i>n</i> – Set to none and clear the hold types which could be applied with the user's level of privilege. Repetition of characters is permitted, but "n" may not appear in the same option argument with the other three characters.</li> </ul> <p>This attribute can be altered once the job has begun execution, but the hold will not take effect unless the job is rerun.</p>
<b>-j</b>	join	<p>Declares which standard streams of the job will be merged together. The join argument value may be the characters "oe" and "eo", or the single character "n".</p> <p>An argument value of oe directs that the standard output and standard error streams of the job will be merged, intermixed, and returned as the standard output. An argument value of eo directs that the standard output and standard error streams of the job will be merged, intermixed, and returned as the standard error.</p> <p>A value of n directs that the two streams will be two separate files. This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> If using either the <code>-e</code> or the <code>-o</code> option and the <code>-j eo oe</code> option, the <code>-j</code> option takes precedence and all standard error and output messages go to the chosen output file.</p> </div>

Option	Name	Description
<b>-k</b>	keep	<p>Defines which if either of standard output or standard error of the job will be retained on the execution host. If set for a stream, this option overrides the path name for that stream.</p> <p>The argument is either the single letter "e", "o", or "n", or one or more of the letters "e" and "o" combined in either order.</p> <ul style="list-style-type: none"> <li>• <i>n</i> – No streams are to be retained.</li> <li>• <i>e</i> – The standard error stream is to be retained on the execution host. The stream will be placed in the home directory of the user under whose user id the job executed. The file name will be the default file name given by:  <code>job_name.e<sub>sequence</sub></code>            where <code>job_name</code> is the name specified for the job, and <code>sequence</code> is the sequence number component of the job identifier.</li> <li>• <i>o</i> – The standard output stream is to be retained on the execution host. The stream will be placed in the home directory of the user under whose user id the job executed. The file name will be the default file name given by:  <code>job_name.o<sub>sequence</sub></code>            where <code>job_name</code> is the name specified for the job, and <code>sequence</code> is the sequence number component of the job identifier.</li> <li>• <i>eo</i> – Both the standard output and standard error streams will be retained.</li> <li>• <i>oe</i> – Both the standard output and standard error streams will be retained.</li> </ul> <p>This attribute cannot be altered once the job has begun execution.</p>
<b>-l</b>	resource_ list	<p>Modifies the list of resources that are required by the job. The <code>resource_list</code> argument is in the following syntax:</p> <pre>resource_name [= [value]] [, resource_name [= [value]], ... ]</pre> <p>For the complete list of resources that can be modified, see <a href="#">Requesting resources on page 2586</a>.</p> <p>If a requested modification to a resource would exceed the resource limits for jobs in the current queue, the server will reject the request.</p> <p>If the job is running, only certain resources can be altered. Which resources can be altered in the run state is system dependent. A user may only lower the limit for those resources.</p>
<b>-m</b>	mail_ options	<p>Replaces the set of conditions under which the execution server will send a mail message about the job. The <code>mail_options</code> argument is a string which consists of the single character "n", or one or more of the characters "a", "b", and "e".</p> <p>If the character "n" is specified, no mail will be sent.</p> <p>For the letters "a", "b", and "e":</p> <ul style="list-style-type: none"> <li>• <i>a</i> – Mail is sent when the job is aborted by the batch system.</li> <li>• <i>b</i> – Mail is sent when the job begins execution.</li> <li>• <i>e</i> – Mail is sent when the job ends.</li> </ul>

Option	Name	Description
<b>-M</b>	user_list	Replaces the list of users to whom mail is sent by the execution server when it sends mail about the job. The user_list argument is of the form: user[@host] [, user[@host], ...]
<b>-n</b>	node-exclusive	Sets or unsets exclusive node allocation on a job. Use the y and n options to enable or disable the feature. This affects only cpusets and compatible schedulers. <pre>&gt; qalter ... -n y #enables exclusive node allocation on a job &gt; qalter ... -n n #disables exclusive node allocation on a job</pre>
<b>-N</b>	name	Renames the job. The name specified may be up to and including 15 characters in length. It must consist of printable, nonwhite space characters with the first character alphabetic.
<b>-o</b>	path	Replaces the path to be used for the standard output stream of the batch job. The path argument is of the form: [hostname:]path_name where <i>hostname</i> is the name of a host to which the file will be returned and <i>path_name</i> is the path name on that host in the syntax recognized by POSIX. The argument will be interpreted as follows: <ul style="list-style-type: none"> <li><i>path_name</i> – Where <i>path_name</i> is not an absolute path name, then the <code>qalter</code> command will expand the path name relative to the current working directory of the command. The command will supply the name of the host upon which it is executing for the <i>hostname</i> component.</li> <li><i>hostname:path_name</i> – Where <i>path_name</i> is not an absolute path name, then the <code>qalter</code> command will not expand the path name. The execution server will expand it relative to the home directory of the user on the system specified by <i>hostname</i>.</li> </ul> This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.
<b>-p</b>	priority	Replaces the priority of the job. The priority argument must be an integer between -1024 and +1023 inclusive. This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.
<b>-r</b>	[y/n]	Declares whether the job is rerunable (see the <code>qrerun</code> command). The option argument <i>c</i> is a single character. PBS recognizes the following characters: y and n. If the argument is "y", the job is marked rerunable. If the argument is "n", the job is marked as not rerunable.

Option	Name	Description
<b>-S</b>	path	<p>Declares the shell that interprets the job script.</p> <p>The option argument path_list is in the form:</p> <pre>path[@host] [,path[@host], ...]</pre> <p>Only one path may be specified for any host named. Only one path may be specified without the corresponding host name. The path selected will be the one with the host name that matched the name of the execution host. If no matching host is found, then the path specified (without a host) will be selected.</p> <p>If the <code>-S</code> option is not specified, the option argument is the null string, or no entry from the path_list is selected, the execution will use the login shell of the user on the execution host.</p> <p>This attribute can be altered once the job has begun execution, but it will not take effect unless the job is rerun.</p>
<b>-t</b>	array_range	<p>The array_range argument is an integer id or a range of integers. Multiple ids or id ranges can be combined in a comma delimited list. Examples: <code>-t 1-100</code> or <code>-t 1, 10, 50-100</code></p> <p>If an array range isn't specified, the command tries to operate on the entire array. The command acts on the array (or specified range of the array) just as it would on an individual job.</p> <p>An optional "slot limit" can be specified to limit the amount of jobs that can run concurrently in the job array. The default value is unlimited. The slot limit must be the last thing specified in the array_request and is delimited from the array by a percent sign (%).</p> <pre>qalter weatherSimulationArray[] -t %20</pre> <p>Here, the array weatherSimulationArray[] is configured to allow a maximum of 20 concurrently running jobs.</p> <p>Slot limits can be applied at job submit time with <a href="#">qsub</a>, or can be set in a global server parameter policy with <a href="#">max_slot_limit</a>.</p>
<b>-u</b>	user_list	<p>Replaces the user name under which the job is to run on the execution system.</p> <p>The user_list argument is of the form:</p> <pre>user[@host] [,user[@host], ...]</pre> <p>Only one user name may be given for per specified host. Only one of the user specifications may be supplied without the corresponding host specification. That user name will be used for execution on any host not named in the argument list.</p> <p>This attribute cannot be altered once the job has begun execution.</p>
<b>-W</b>	additional_attributes	<p>The <code>-w</code> option allows for the modification of additional job attributes.</p> <p>Note if white space occurs anywhere within the option argument string or the equal sign, "=", occurs within an attribute_value string, then the string must be enclosed with either single or double quote marks.</p> <p>To see the attributes PBS currently supports within the <code>-w</code> option, see <a href="#">Table 4-5: -W additional_attributes on page 2716</a>.</p>

Table 4-5: -W additional\_attributes

Attribute	Description
<b>depend=dependency_list</b>	<p>Redefines the dependencies between this and other jobs. The <code>dependency_list</code> is in the form:</p> <pre>type [:argument[:argument...]] [, type:argument...]</pre> <p>The argument is either a numeric count or a PBS job id according to type. If argument is a count, it must be greater than 0. If it is a job id and is not fully specified in the form: <code>seq_number.server.name</code>, it will be expanded according to the default server rules. If argument is null (the preceding colon need not be specified), the dependency of the corresponding type is cleared (unset).</p> <ul style="list-style-type: none"> <li>• <i>synccount:count</i> – This job is the first in a set of jobs to be executed at the same time. Count is the number of additional jobs in the set.</li> <li>• <i>syncwith:jobid</i> – This job is an additional member of a set of jobs to be executed at the same time. In the above and following dependency types, jobid is the job identifier of the first job in the set.</li> <li>• <i>after:jobid [:jobid...]</i> – This job may be scheduled for execution at any point after jobs jobid have started execution.</li> <li>• <i>afterok:jobid [:jobid...]</i> – This job may be scheduled for execution only after jobs jobid have terminated with no errors. See the csh warning under "Extended Description".</li> <li>• <i>afternotok:jobid [:jobid...]</i> – This job may be scheduled for execution only after jobs jobid have terminated with errors. See the csh warning under "Extended Description".</li> <li>• <i>afterany:jobid [:jobid...]</i> – This job may be scheduled for execution after jobs jobid have terminated, with or without errors.</li> <li>• <i>on:count</i> – This job may be scheduled for execution after count dependencies on other jobs have been satisfied. This dependency is used in conjunction with any of the 'before' dependencies shown below. If job A has <i>on:2</i>, it will wait for two jobs with 'before' dependencies on job A to be fulfilled before running.</li> <li>• <i>before:jobid [:jobid...]</i> – When this job has begun execution, then jobs jobid... may begin.</li> <li>• <i>beforeok:jobid [:jobid...]</i> – If this job terminates execution without errors, then jobs jobid... may begin. See the csh warning under "Extended Description".</li> <li>• <i>beforenotok:jobid [:jobid...]</i> – If this job terminates execution with errors, then jobs jobid... may begin. See the csh warning under "Extended Description".</li> <li>• <i>beforeany:jobid [:jobid...]</i> – When this job terminates execution, jobs jobid... may begin.</li> </ul> <p>If any of the before forms are used, the job referenced by jobid must have been submitted with a dependency type of on.</p> <p>If any of the before forms are used, the jobs referenced by jobid must have the same owner as the job being altered. Otherwise, the dependency will not take effect.</p> <p>Error processing of the existence, state, or condition of the job specified to qalter is a deferred service, i.e. the check is performed after the job is queued. If an error is detected, the job will be deleted by the server. Mail will be sent to the job submitter stating the error.</p>

Attribute	Description
<b>group_list=g_list</b>	<p>Alters the group name under which the job is to run on the execution system.</p> <p>The <code>g_list</code> argument is of the form:</p> <pre>group[@host] [,group[@host], ...]</pre> <p>Only one group name may be given per specified host. Only one of the group specifications may be supplied without the corresponding host specification. That group name will be used for execution on any host not named in the argument list.</p>
<b>stagein=file_list</b> <b>stageout=file_list</b>	<p>Alters which files are staged (copied) in before job start or staged out after the job completes execution. The <code>file_list</code> is in the form:</p> <pre>local_file@hostname:remote_file[,...]</pre> <p>The name <code>local_file</code> is the name on the system where the job executes. It may be an absolute path or a path relative to the home directory of the user. The name <code>remote_file</code> is the destination name on the host specified by <code>hostname</code>. The name may be absolute or relative to the user's home directory on the destination host.</p>

## Operands

The `qalter` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name][@server]
```

## Standard error

Any error condition, either in processing the options or the operands, or any error received in reply to the batch requests will result in an error message being written to standard error.

## Exit status

Upon successful processing of all the operands presented to the `qalter` command, the exit status will be a value of zero.

If the `qalter` command fails to process any operand, the command exits with a value greater than zero.

## Copyright

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1, 2003 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6, Copyright © 2001-2003 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

## Related topics

- [qdel](#)
- [qhold](#)
- [qrls](#)
- [qsub](#)

## Non-Adaptive Computing topics

- Batch Environment Services
- [qmove](#)
- [touch](#)

## qchkpt

Checkpoint pbs batch jobs.

## Synopsis

```
qchkpt <JOBID> [ <JOBID> ] ...
```

## Description

The `qchkpt` command requests that the PBS MOM generate a checkpoint file for a running job.

This is an extension to POSIX.2d.

The `qchkpt` command sends a Chkpt Job batch request to the server as described in the general section.

## Options

None.

## Operands

The `qchkpt` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name] [@server]
```

## Examples

```
> qchkpt 3233 request a checkpoint for job 3233
```

## Standard error

The `qchkpt` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qchkpt` command, the exit status will be a value of zero.

If the `qchkpt` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qhold\(1B\)](#)
- [qrls\(1B\)](#)
- [qalter\(1B\)](#)
- [qsub\(1B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_alterjob\(3B\)](#)
- [pbs\\_holdjob\(3B\)](#),
- [pbs\\_rlsjob\(3B\)](#)
- [pbs\\_job\\_attributes\(7B\)](#)
- [pbs\\_resources\\_unicos8\(7B\)](#)

## qdel

*(delete job)*

### Synopsis

```
qdel [{-a <asynchronous delete>|-m <message>|-p|-W <delay>|-t <array_range>}]
<JOBID>[ <JOBID>]... | 'all' | 'ALL'
```

### Description

The `qdel` command deletes jobs in the order in which their job identifiers are presented to the command. A job is deleted by sending a Delete Job batch request to the batch server that owns the job. A job that has been deleted is no longer subject to management by batch services.

A batch job may be deleted by its owner, the batch operator, or the batch administrator.

A batch job being deleted by a server will be sent a SIGTERM signal following by a SIGKILL signal. The time delay between the two signals is an attribute of the execution queue from which the job was run (set table by the administrator). This delay may be overridden by the `-W` option.

See the PBS ERS section 3.1.3.3, "Delete Job Request", for more information.

## Options

Option	Name	Description
<b>-a</b>	asynchronous delete	Performs an asynchronous delete. The server responds to the user before contacting the MOM. The option <code>qdel -a all</code> performs <code>qdel all</code> due to restrictions from being single-threaded.
<b>-W</b>	delay	Specifies the wait delay between the sending of the SIGTERM and SIGKILL signals. The argument is the length of time in seconds of the delay.
<b>-p</b>	purge	Forcibly purges the job from the server. This should only be used if a running job will not exit because its allocated nodes are unreachable. The admin should make every attempt at resolving the problem on the nodes. If a job's mother superior recovers after purging the job, any epilogue scripts may still run. This option is only available to a batch operator or the batch administrator.
<b>-m</b>	message	Specify a comment to be included in the email. The argument message specifies the comment to send. This option is only available to a batch operator or the batch administrator.
<b>-t</b>	array_range	The <code>array_range</code> argument is an integer id or a range of integers. Multiple ids or id ranges can be combined in a comma delimited list (examples: <code>-t 1-100</code> or <code>-t 1,10,50-100</code> ).  If an array range isn't specified, the command tries to operate on the entire array. The command acts on the array (or specified range of the array) just as it would on an individual job.

## Operands

The `qdel` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name] [@server]
```

or

```
all
```

## Examples

```
> qdel 1324
> qdel 1324-3 To delete one job of a job array
> qdel all To delete all jobs (Version 2.3.0 and later)
```

## Standard error

The `qdel` command will write a diagnostic messages to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qdel` command, the exit status will be a value of zero.

If the `qdel` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qsub\(1B\)](#)
- [qsig\(1B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_deljob\(3B\)](#)

## qgpumode

*(GPU mode)*

## Synopsis

```
qgpumode -H host -g gpuid -m mode
```

## Description

The `qgpumode` command specifies the mode for the GPU. This command triggers an immediate update of the `pbs_server`.



For additional information about options for configuring GPUs, see [NVIDIA GPUs on page 817](#).

## Options

Option	Description
<b>-H</b>	Specifies the host where the GPU is located.
<b>-g</b>	Specifies the ID of the GPU. This varies depending on the version of the Nvidia driver used. For driver 260.x, it is 0, 1, and so on. For driver 270.x, it is the PCI bus address, i.e., 0:5:0.

Option	Description
<b>-m</b>	<p>Specifies the new mode for the GPU:</p> <ul style="list-style-type: none"> <li>• <b>0 (Default/Shared)</b>: Default/shared compute mode. Multiple threads can use <code>cudaSetDevice()</code> with this device.</li> <li>• <b>1 (Exclusive Thread)</b>: Compute-exclusive-thread mode. Only one thread in one process is able to use <code>cudaSetDevice()</code> with this device.</li> <li>• <b>2 (Prohibited)</b>: Compute-prohibited mode. No threads can use <code>cudaSetDevice()</code> with this device.</li> <li>• <b>3 (Exclusive Process)</b>: Compute-exclusive-process mode. Many threads in one process are able to use <code>cudaSetDevice()</code> with this device.</li> </ul> <pre>qgpumode -H node01 -g 0 -m 1</pre> <p><i>This puts the first GPU on node01 into mode 1 (exclusive)</i></p> <pre>qgpumode -H node01 -g 0 -m 0</pre> <p><i>This puts the first GPU on node01 into mode 0 (shared)</i></p>

#### Related topics

- [qgpureset on page 2723](#)

## qgpureset

(reset GPU)

### Synopsis

```
qgpureset -H host -g gpuid -p -v
```

### Description

The `qgpureset` command resets the GPU.

### Options

Option	Description
<b>-H</b>	Specifies the host where the GPU is located.
<b>-g</b>	Specifies the ID of the GPU. This varies depending on the version of the Nvidia driver used. For driver 260.x, it is 0, 1, and so on. For driver 270.x, it is the PCI bus address, i.e., 0:5:0.

Option	Description
<b>-p</b>	Specifies to reset the GPU's permanent ECC error count.
<b>-v</b>	Specifies to reset the GPU's volatile ECC error count.

## Related topics

- [ggpumode on page 2722](#)

## qhold

*(hold job)*

## Synopsis

```
qhold [{"-h" <HOLD LIST>|"-t" <array_range>}] <JOBID>[ <JOBID>] ...
```

## Description

The `qhold` command requests that the server place one or more holds on a job. A job that has a hold is not eligible for execution. There are three supported holds: USER, OTHER (also known as operator), and SYSTEM.

A user may place a USER hold upon any job the user owns. An "operator", who is a user with "operator privilege," may place either an USER or an OTHER hold on any job. The batch administrator may place any hold on any job.

If no `-h` option is given, the USER hold will be applied to the jobs described by the `job_identifier` operand list.

If the job identified by `job_identifier` is in the queued, held, or waiting states, then the hold type is added to the job. The job is then placed into held state if it resides in an execution queue.

If the job is in running state, then the following additional action is taken to interrupt the execution of the job. If checkpoint/restart is supported by the host system, requesting a hold on a running job will (1) cause the job to be checkpointed, (2) the resources assigned to the job will be released, and (3) the job is placed in the held state in the execution queue.

If checkpoint/restart is not supported, `qhold` will only set the requested hold attribute. This will have no effect unless the job is rerun with the [qrerun](#) command.

## Options

Option	Name	Description
<b>-h</b>	hold_list	The hold_list argument is a string consisting of one or more of the letters "u", "o", or "s" in any combination. The hold type associated with each letter is: <ul style="list-style-type: none"> <li>• <i>u</i> - USER</li> <li>• <i>o</i> - OTHER</li> <li>• <i>s</i> - SYSTEM</li> </ul>
<b>-t</b>	array_range	The array_range argument is an integer id or a range of integers. Multiple ids or id ranges can be combined in a comma delimited list (examples: -t 1-100 or -t 1,10,50-100) . If an array range isn't specified, the command tries to operate on the entire array. The command acts on the array (or specified range of the array) just as it would on an individual job.

## Operands

The `qhold` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name] [@server]
```

## Example

```
> qhold -h u 3233 place user hold on job 3233
```

## Standard error

The `qhold` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qhold` command, the exit status will be a value of zero.

If the `qhold` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qrls\(1B\)](#)
- [qalter\(1B\)](#)
- [qsub\(1B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_alterjob\(3B\)](#)
- [pbs\\_holdjob\(3B\)](#)

- pbs\_rlsjob(3B)
- pbs\_job\_attributes(7B)
- pbs\_resources\_unicos8(7B)

## qmgr

(PBS Queue Manager) PBS batch system manager.

### Synopsis

```
qmgr [-a] [-c command] [-e] [-n] [-z] [server...]
```

### Description

The `qmgr` command provides an administrator interface to query and configure batch system parameters (see [Appendix B: Server parameters on page 2766](#)).

The command reads directives from standard input. The syntax of each directive is checked and the appropriate request is sent to the batch server or servers.

The list or print subcommands of `qmgr` can be executed by general users. Creating or deleting a queue requires PBS Manager privilege. Setting or unsetting server or queue attributes requires PBS Operator or Manager privilege.

**i** By default, the user `root` is the only PBS Operator and Manager. To allow other users to be privileged, the server attributes `operators` and `managers` will need to be set (i.e., as `root`, issue `'qmgr -c 'set server managers += <USER1>@<HOST>'`). See [TORQUE/PBS Integration Guide - RM Access Control on page 1231](#).

If `qmgr` is invoked without the `-c` option and standard output is connected to a terminal, `qmgr` will write a prompt to standard output and read a directive from standard input.

Commands can be abbreviated to their minimum unambiguous form. A command is terminated by a new line character or a semicolon, ";", character. Multiple commands may be entered on a single line. A command may extend across lines by escaping the new line character with a back-slash "\".

Comments begin with the "#" character and continue to end of the line. Comments and blank lines are ignored by `qmgr`.

### Options

Option	Name	Description
<code>-a</code>	---	Abort <code>qmgr</code> on any syntax errors or any requests rejected by a server.
<code>-c</code>	command	Execute a single command and exit <code>qmgr</code> .

Option	Name	Description
<b>-e</b>	---	Echo all commands to standard output.
<b>-n</b>	---	No commands are executed, syntax checking only is performed.
<b>-z</b>	---	No errors are written to standard error.

## Operands

The *server* operands identify the name of the batch server to which the administrator requests are sent. Each *server* conforms to the following syntax:

```
host_name[:port]
```

where *host\_name* is the network name of the host on which the server is running and *port* is the port number to which to connect. If *port* is not specified, the default port number is used.

If *server* is not specified, the administrator requests are sent to the local server.

## Standard input

The `qmgr` command reads standard input for directives until end of file is reached, or the exit or quit directive is read.

## Standard output

If Standard Output is connected to a terminal, a command prompt will be written to standard output when `qmgr` is ready to read a directive.

If the `-e` option is specified, `qmgr` will echo the directives read from standard input to standard output.

## Standard error

If the `-z` option is not specified, the `qmgr` command will write a diagnostic message to standard error for each error occurrence.

## Directive syntax

A `qmgr` directive is one of the following forms:

```
command server [names] [attr OP value[,attr OP value,...]]
command queue [names] [attr OP value[,attr OP value,...]]
command node [names] [attr OP value[,attr OP value,...]]
```

where *command* is the command to perform on an object.

Commands are:

Command	Description
<b>active</b>	Sets the active objects. If the active objects are specified, and the name is not given in a <code>qmgr cmd</code> the active object names will be used.
<b>create</b>	Is to create a new object, applies to queues and nodes.
<b>delete</b>	Is to destroy an existing object, applies to queues and nodes.
<b>set</b>	Is to define or alter attribute values of the object.
<b>unset</b>	Is to clear the value of attributes of the object. <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;">  This form does not accept an OP and value, only the attribute name. </div>
<b>list</b>	Is to list the current attributes and associated values of the object.
<b>print</b>	Is to print all the queue and server attributes in a format that will be usable as input to the <code>qmgr</code> command.
<b>names</b>	Is a list of one or more names of specific objects The name list is in the form: <code>[name] [@server] [, queue_name [@server] . . . ]</code> with no intervening white space. The name of an object is declared when the object is first created. If the name is <code>@server</code> , then all the objects of specified type at the server will be affected.
<b>attr</b>	Specifies the name of an attribute of the object which is to be set or modified. If the attribute is one which consist of a set of resources, then the attribute is specified in the form: <code>attribute_name.resource_name</code>
<b>OP</b>	Operation to be performed with the attribute and its value: <ul style="list-style-type: none"> <li>• "=" – set the value of the attribute. If the attribute has an existing value, the current value is replaced with the new value.</li> <li>• "+=" – increase the current value of the attribute by the amount in the new value.</li> <li>• "-=" – decrease the current value of the attribute by the amount in the new value.</li> </ul>
<b>value</b>	The value to assign to an attribute. If the value includes white space, commas or other special characters, such as the "#" character, the value string must be enclosed in quote marks ("").

The following are examples of `qmgr` directives:

```

create queue fast priority=10,queue_type=e,enabled = true,max_running=0
set queue fast max_running +=2
create queue little
set queue little resources_max.mem=8mw,resources_max.cput=10
unset queue fast max_running
set node state = "down,offline"
active server s1,s2,s3
list queue @server1
set queue max_running = 10          - uses active queues

```

## Exit status

Upon successful processing of all the operands presented to the `qmgr` command, the exit status will be a value of zero.

If the `qmgr` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [pbs\\_server\(8B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_queue\\_attributes\(7B\)](#)
- [pbs\\_server\\_attributes\(7B\)](#)
- [qstart\(8B\)](#), [qstop\(8B\)](#)
- [qenable\(8B\)](#), [qdisable\(8\)](#)
- [PBS External Reference Specification](#)

## qmove

Move PBS batch jobs.

## Synopsis

```
qmove destination jobId [jobId ...]
```

## Description

To move a job is to remove the job from the queue in which it resides and instantiate the job in another queue. The `qmove` command issues a Move Job batch request to the batch server that currently owns each job specified by *jobId*.

A job in the **Running**, **Transiting**, or **Exiting** state cannot be moved.

## Operands

The first operand, the new *destination*, is one of the following:

queue

@server

queue@server

If the *destination* operand describes only a queue, then `qmove` will move jobs into the queue of the specified name at the job's current server. If the *destination* operand describes only a batch server, then `qmove` will move jobs into the default queue at that batch server. If the *destination* operand describes both a queue and a batch server, then `qmove` will move the jobs into the specified queue at the specified server.

All following operands are *joblds* which specify the jobs to be moved to the new *destination*. The `qmove` command accepts one or more *jobld* operands of the form: `sequenceNumber [.serverName] [@server]`

## Standard error

The `qmove` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qmove` command, the exit status will be a value of zero.

If the `qmove` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qsub on page 2744](#)

### Related topics(non-Adaptive Computing topics)

- [pbs\\_movejob\(3B\)](#)

## qorder

Exchange order of two PBS batch jobs in any queue.

## Synopsis

```
qorder job1_identifier job2_identifier
```

## Description

To order two jobs is to exchange the jobs' positions in the queue(s) in which the jobs reside. The two jobs must be located on the same server. No attribute of the job, such as priority, is changed. The impact of changing the order in the queue(s) is dependent on local job schedule policy. For information about your local job schedule policy, contact your systems administrator.

 A job in the **running** state cannot be reordered.

## Operands

Both operands are `job_identifiers` that specify the jobs to be exchanged. The `qorder` command accepts two `job_identifier` operands of the following form:  
`sequence_number [.server_name] [@server]`

The two jobs must be in the same location, so the server specification for the two jobs must agree.

## Standard error

The `qorder` command will write diagnostic messages to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qorder` command, the exit status will be a value of zero.

If the `qorder` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qsub on page 2744](#)
- [qmove on page 2729](#)

### Related topics(non-Adaptive Computing topics)

- [pbs\\_orderjob\(3B\)](#)
- [pbs\\_movejob\(3B\)](#)

## qrerun

*(Rerun a batch job)*

## Synopsis

```
qrerun [{-f}] <JOBID>[ <JOBID>] ...
```

## Description

The `qrerun` command directs that the specified jobs are to be rerun if possible. To rerun a job is to terminate the session leader of the job and return the job to the queued state in the execution queue in which the job currently resides.

If a job is marked as not rerunable then the rerun request will fail for that job. If the mini-server running the job is down, or it rejects the request, the Rerun Job batch request will return a failure unless `-f` is used.

Using `-f` violates IEEE Batch Processing Services Standard and should be handled with great care. It should only be used under exceptional circumstances. The best practice is to fix the problem mini-server host and let `qrerun` run normally. The nodes may need manual cleaning (see the `-r` option on the [qsub](#) and [qalter](#) commands).

## Options

Option	Description
<b>-f</b>	Force a rerun on a job

```
qrerun -f 15406
```

**i** The `qrerun` all command is meant to be run if all of the compute nodes go down. If the machines have actually crashed, then we know that all of the jobs need to be restarted. The behavior if you don't run this would depend on how you bring up the `pbs_mom` daemons, but by default would be to cancel all of the jobs.

Running the command makes it so that all jobs are requeued without attempting to contact the moms on which they should be running.

## Operands

The `qrerun` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name][@server]
```

## Standard error

The `qrerun` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qrerun` command, the exit status will be a value of zero.

If the `qrerun` command fails to process any operand, the command exits with a value greater than zero.

## Examples

```
> qrerun 3233
```

(Job 3233 will be re-run.)

### Related topics

- [qsub\(1B\)](#)
- [qalter\(1B\)](#)

## Non-Adaptive Computing topics

- [pbs\\_alterjob\(3B\)](#)
- [pbs\\_rerunjob\(3B\)](#)

## qrlls

*(Release hold on PBS batch jobs)*

## Synopsis

```
qrlls [{"-h" <HOLD LIST>|"-t" <array_range>}] <JOBID>[ <JOBID>] ...
```

## Description

The `qrlls` command removes or releases holds which exist on batch jobs.

A job may have one or more types of holds which make the job ineligible for execution. The types of holds are USER, OTHER, and SYSTEM. The different types of holds may require that the user issuing the `qrlls` command have special privileges. A user may always remove a USER hold on their own jobs, but only privileged users can remove OTHER or SYSTEM holds. An attempt to release a hold for which the user does not have the correct privilege is an error and no holds will be released for that job.

If no `-h` option is specified, the USER hold will be released.

If the job has no execution\_time pending, the job will change to the queued state. If an execution\_time is still pending, the job will change to the waiting state.

## Options

Command	Name	Description
<code>-h</code>	hold_list	<p>Defines the types of hold to be released from the jobs. The hold_list option argument is a string consisting of one or more of the letters "u", "o", and "s" in any combination. The hold type associated with each letter is:</p> <ul style="list-style-type: none"> <li>• <i>u</i> - USER</li> <li>• <i>o</i> - OTHER</li> <li>• <i>s</i> - SYSTEM</li> </ul>
<code>-t</code>	array_range	<p>The array_range argument is an integer id or a range of integers. Multiple ids or id ranges can be combined in a comma delimited list. Examples: <code>-t 1-100</code> or <code>-t 1,10,50-100</code></p> <p>If an array range isn't specified, the command tries to operate on the entire array. The command acts on the array (or specified range of the array) just as it would on an individual job.</p>

## Operands

The `qrls` command accepts one or more `job_identifier` operands of the form:

```
sequence_number [.server_name] [@server]
```

## Examples

```
> qrls -h u 3233 release user hold on job 3233
```

## Standard error

The `qrls` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qrls` command, the exit status will be a value of zero.

If the `qrls` command fails to process any operand, the command exits with a value greater than zero.

Related topics

Related topics

- [qsub\(1B\)](#)
- [qalter\(1B\)](#)
- [qhold\(1B\)](#)

## Non-Adaptive Computing topics)

- [pbs\\_alterjob\(3B\)](#)
- [pbs\\_holdjob\(3B\)](#)
- [pbs\\_rlsjob\(3B\)](#)

## qrun

*(Run a batch job)*

## Synopsis

```
qrun [{"-H <HOST>|_a"}] <JOBID>[ <JOBID>] ...
```

## Overview

The `qrun` command runs a job.

## Format

-H	
<b>Format</b>	<STRING> Host Identifier
<b>Default</b>	---
<b>Description</b>	Specifies the host within the cluster on which the job(s) are to be run. The host argument is the name of a host that is a member of the cluster of hosts managed by the server. If the option is not specified, the server will select the "worst possible" host on which to execute the job.
<b>Example</b>	<pre>qrun -H hostname 15406</pre>

-a	
<b>Format</b>	---
<b>Default</b>	---
<b>Description</b>	Run the job(s) asynchronously.
<b>Example</b>	<pre>qrun -a 15406</pre>

## Command details

The `qrun` command is used to force a batch server to initiate the execution of a batch job. The job is run regardless of scheduling position or resource requirements.

In order to execute `qrun`, the user must have PBS Operation or Manager privileges.

## Examples

```
> qrun 3233
```

(Run job 3233.)

`qsig`

(Signal a job)

## Synopsis

```
qsig [{-s <SIGNAL>}] <JOBID>[ <JOBID>] ...
[-a]
```

## Description

The `qsig` command requests that a signal be sent to executing batch jobs. The signal is sent to the session leader of the job. If the `-s` option is not specified, `SIGTERM` is sent. The request to signal a batch job will be rejected if:

- The user is not authorized to signal the job.
- The job is not in the running state.
- The requested signal is not supported by the system upon which the job is executing.

The `qsig` command sends a Signal Job batch request to the server which owns the job.

## Options

Option	Name	Description
<code>-s</code>	signal	<p>Declares which signal is sent to the job.</p> <p>The signal argument is either a signal name, e.g. <code>SIGKILL</code>, the signal name without the <code>SIG</code> prefix, e.g. <code>KILL</code>, or an unsigned signal number, e.g. <code>9</code>. The signal name <code>SIGNULL</code> is allowed; the server will send the signal <code>0</code> to the job which will have no effect on the job, but will cause an obituary to be sent if the job is no longer executing. Not all signal names will be recognized by <code>qsig</code>. If it doesn't recognize the signal name, try issuing the signal number instead.</p> <p>Two special signal names, "suspend" and "resume", are used to suspend and resume jobs. Cray systems use the Cray-specific <code>suspend()/resume()</code> calls.</p> <p>On non-Cray system, <code>suspend</code> causes a <code>SIGTSTP</code> to be sent to all processes in the job's top task, wait 5 seconds, and then send a <code>SIGSTOP</code> to all processes in all tasks on all nodes in the job. This differs from TORQUE 2.0.0 which did not have the ability to propagate signals to sister nodes. <code>Resume</code> sends a <code>SIGCONT</code> to all processes in all tasks on all nodes.</p> <p>When suspended, a job continues to occupy system resources but is not executing and is not charged for walltime. The job will be listed in the "S" state. Manager or operator privilege is required to suspend or resume a job.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p> Interactive jobs may not resume properly because the top-level shell will background the suspended child process.</p> </div>
<code>-a</code>	asynchronously	Makes the command run asynchronously.

## Operands

The `qsig` command accepts one or more `job_identifier` operands of the form:

```
sequence_number[.server_name] [@server]
```

## Examples

```
> qsig -s SIGKILL 3233      send a SIGKILL to job 3233
> qsig -s KILL 3233        send a SIGKILL to job 3233
> qsig -s 9 3233           send a SIGKILL to job 3233
```

## Standard error

The `qsig` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qsig` command, the exit status will be a value of zero.

If the `qsig` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qsub\(1B\)](#)

### Non-Adaptive Computing topics

- [pbs\\_sigjob\(3B\)](#)
- [pbs\\_resources\\_\\*\(7B\)](#) where \* is system type
- PBS ERS

## qstat

Show status of PBS batch jobs.

## Synopsis

```
qstat [-c on page 2738] [-f [-1]] [-W site_specific] [job_identifier... |
destination...] [time]
qstat [-a|-i|-r|-e] [-c on page 2738] [-n [-1]] [-s] [-G|-M] [-R] [-u user_
list]
[job_identifier... | destination...]
qstat -Q [-f [-1]] [-c on page 2738] [-W site_specific] [destination...]
qstat -q [-c on page 2738] [-G|-M] [destination...]
qstat -B [-c on page 2738] [-f [-1]] [-W site_specific] [server_name...]
qstat -t [-c on page 2738]
```

## Description

The `qstat` command is used to request the status of jobs, queues, or a batch server. The requested status is written to standard out.

When requesting job status, synopsis format 1 or 2, `qstat` will output information about each job\_ identifier or all jobs at each destination. Jobs for which the user does not have status privilege are not displayed.

When requesting queue or server status, synopsis format 3 through 5, `qstat` will output information about each destination.

**i** You can configure TORQUE with `CFLAGS='DTXT'` to change the alignment of text in `qstat` output. This noticeably improves `qstat -r` output.

## Options

Option	Description
<b>-c</b>	Completed jobs are not displayed in the output. If desired, you can set the <code>PBS_QSTAT_NO_COMPLETE</code> environment variable to cause all <code>qstat</code> requests to not show completed jobs by default.
<b>-f</b>	Specifies that a full status display be written to standard out. The [time] value is the amount of wall-time, in seconds, remaining for the job. [time] does not account for walltime multipliers.
<b>-a</b>	All jobs are displayed in the alternative format (see <a href="#">Standard output on page 2740</a> ). If the operand is a destination id, all jobs at that destination are displayed. If the operand is a job id, information about that job is displayed.
<b>-e</b>	If the operand is a job id or not specified, only jobs in executable queues are displayed. Setting the <code>PBS_QSTAT_EXEONLY</code> environment variable will also enable this option.
<b>-i</b>	Job status is displayed in the alternative format. For a destination id operand, statuses for jobs at that destination which are not running are displayed. This includes jobs which are queued, held or waiting. If an operand is a job id, status for that job is displayed regardless of its state.
<b>-r</b>	If an operand is a job id, status for that job is displayed. For a destination id operand, statuses for jobs at that destination which are running are displayed; this includes jobs which are suspended. Note that if there is no walltime given for a job, then elapsed time does not display.
<b>-n</b>	In addition to the basic information, nodes allocated to a job are listed.
<b>-1</b>	In combination with <a href="#">-n</a> , the <code>-1</code> option puts all of the nodes on the same line as the job ID. In combination with <a href="#">-f</a> attributes are not folded to fit in a terminal window. This is intended to ease the parsing of the <code>qstat</code> output.
<b>-s</b>	In addition to the basic information, any comment provided by the batch administrator or scheduler is shown.

Option	Description
<b>-G</b>	Show size information in giga-bytes.
<b>-M</b>	Show size information, disk or memory in mega-words. A word is considered to be 8 bytes.
<b>-R</b>	In addition to other information, disk reservation information is shown. Not applicable to all systems.
<b>-t</b>	Normal <code>qstat</code> output displays a summary of the array instead of the entire array, job for job. <code>qstat -t</code> expands the output to display the entire array. Note that arrays are now named with brackets following the array name; for example: <pre>dbbeer@napali:~/dev/torque/array_changes\$ echo sleep 20   qsub -t 0-299 189 []].napali</pre> Individual jobs in the array are now also noted using square brackets instead of dashes; for example, here is part of the output of <code>qstat -t</code> for the preceding array: <pre>189[299].napali STDIN[299] dbbeer 0 Q batch</pre>
<b>-u</b>	Job status is displayed in the alternative format. If an operand is a job id, status for that job is displayed. For a destination id operand, statuses for jobs at that destination which are owned by the user(s) listed in <code>user_list</code> are displayed. The syntax of the <code>user_list</code> is: <pre>user_name[@host] [, user_name[@host], ...]</pre> Host names may be wild carded on the left end, e.g. <code>*.nasa.gov</code> . User_name without a <code>@host</code> is equivalent to <code>"user_name@*"</code> , that is at any host.
<b>-Q</b>	Specifies that the request is for queue status and that the operands are destination identifiers.
<b>-q</b>	Specifies that the request is for queue status which should be shown in the alternative format.
<b>-B</b>	Specifies that the request is for batch server status and that the operands are the names of servers.

## Operands

If neither the **-Q** nor the **-B** option is given, the operands on the `qstat` command must be either job identifiers or destinations identifiers.

If the operand is a job identifier, it must be in the following form:

```
sequence_number[.server_name] [@server]
```

where `sequence_number.server_name` is the job identifier assigned at submittal time (see [qsub](#)). If the `.server_name` is omitted, the name of the default server will be used. If `@server` is supplied, the request will be for the job identifier currently at that Server.

If the operand is a destination identifier, it is one of the following three forms:

- queue
- @server
- queue@server

If queue is specified, the request is for status of all jobs in that queue at the default server. If the @server form is given, the request is for status of all jobs at that server. If a full destination identifier, queue@server, is given, the request is for status of all jobs in the named queue at the named server.

If the **-Q** option is given, the operands are destination identifiers as specified above. If queue is specified, the status of that queue at the default server will be given. If queue@server is specified, the status of the named queue at the named server will be given. If @server is specified, the status of all queues at the named server will be given. If no destination is specified, the status of all queues at the default server will be given.

If the **-B** option is given, the operand is the name of a server.

## Standard output

### Displaying job status

If job status is being displayed in the default format and the **-f** option is not specified, the following items are displayed on a single line, in the specified order, separated by white space:

- the job identifier assigned by PBS.
- the job name given by the submitter.
- the job owner.
- the CPU time used.
- the job state:

Item	Description
<b>C</b>	Job is completed after having run.
<b>E</b>	Job is exiting after having run.
<b>H</b>	Job is held.
<b>Q</b>	Job is queued, eligible to run or routed.
<b>R</b>	Job is running.
<b>T</b>	Job is being moved to new location.

Item	Description
<b>W</b>	Job is waiting for its execution time ( <a href="#">-a</a> option) to be reached.
<b>S</b>	(Unicos only) Job is suspended.

- the queue in which the job resides.

If job status is being displayed and the [-f](#) option is specified, the output will depend on whether `qstat` was compiled to use a Tcl interpreter. See [Configuration on page 2743](#) for details. If Tcl is not being used, full display for each job consists of the header line:

```
Job Id: job identifier
```

Followed by one line per job attribute of the form:

```
attribute_name = value
```

If any of the options [-a](#), [-i](#), [-r](#), [-u](#), [-n](#), [-s](#), [-G](#), or [-M](#) are provided, the alternative display format for jobs is used. The following items are displayed on a single line, in the specified order, separated by white space:

- the job identifier assigned by PBS
- the job owner
- the queue in which the job currently resides
- the job name given by the submitter
- the session id (if the job is running)
- the number of nodes requested by the job
- the number of cpus or tasks requested by the job
- the amount of memory requested by the job
- either the cpu time, if specified, or wall time requested by the job, (hh:mm)
- the jobs current state
- the amount of cpu time or wall time used by the job (hh:mm)

If the [-r](#) option is provided, the line contains:

- the job identifier assigned by PBS
- the job owner
- the queue in which the job currently resides
- the number of nodes requested by the job
- the number of cpus or tasks requested by the job
- the amount of memory requested by the job
- either the cpu time or wall time requested by the job

- the jobs current state
- the amount of cpu time or wall time used by the job
- the amount of SRFS space requested on the big file system
- the amount of SRFS space requested on the fast file system
- the amount of space requested on the parallel I/O file system

The last three fields may not contain useful information at all sites or on all systems

## Displaying queue status

If queue status is being displayed and the `-f` option was not specified, the following items are displayed on a single line, in the specified order, separated by white space:

- the queue name
- the maximum number of jobs that may be run in the queue concurrently
- the total number of jobs in the queue
- the enable or disabled status of the queue
- the started or stopped status of the queue
- for each job state, the name of the state and the number of jobs in the queue in that state
- the type of queue, execution or routing

If queue status is being displayed and the `-f` option is specified, the output will depend on whether `qstat` was compiled to use a Tcl interpreter. See [Configuration on page 2743](#) for details. If Tcl is not being used, the full display for each queue consists of the header line:

```
Queue: queue_name
```

Followed by one line per queue attribute of the form:

```
attribute_name = value
```

If the `-Q` option is specified, queue information is displayed in the alternative format: The following information is displayed on a single line:

- the queue name
- the maximum amount of memory a job in the queue may request
- the maximum amount of cpu time a job in the queue may request
- the maximum amount of wall time a job in the queue may request
- the maximum amount of nodes a job in the queue may request
- the number of jobs in the queue in the running state
- the number of jobs in the queue in the queued state
- the maximum number (limit) of jobs that may be run in the queue concurrently

- the state of the queue given by a pair of letters:
  - either the letter *E* if the queue is Enabled or *D* if Disabled
  - and
  - either the letter *R* if the queue is Running (started) or *S* if Stopped.

### Displaying server status

If batch server status is being displayed and the `-f` option is not specified, the following items are displayed on a single line, in the specified order, separated by white space:

- the server name
- the maximum number of jobs that the server may run concurrently
- the total number of jobs currently managed by the server
- the status of the server
- for each job state, the name of the state and the number of jobs in the server in that state

If server status is being displayed and the `-f` option is specified, the output will depend on whether `qstat` was compiled to use a Tcl interpreter. See [Configuration on page 2743](#) for details. If Tcl is not being used, the full display for the server consists of the header line:

```
Server: server name
```

Followed by one line per server attribute of the form:

```
attribute_name = value
```

### Standard error

The `qstat` command will write a diagnostic message to standard error for each error occurrence.

### Configuration

If `qstat` is compiled with an option to include a Tcl interpreter, using the `-f` flag to get a full display causes a check to be made for a script file to use to output the requested information. The first location checked is `$HOME/.qstatrc`. If this does not exist, the next location checked is administrator configured. If one of these is found, a Tcl interpreter is started and the script file is passed to it along with three global variables. The command line arguments are split into two variable named flags and operands. The status information is passed in a variable named `objects`. All of these variables are Tcl lists. The flags list contains the name of the command (usually "qstat") as its first element. Any other elements are command line option flags with any options they use, presented in the order given on the command line. They are broken up individually so that if two flags are given together on the command line, they are separated in the list. For example, if the user typed:

```
qstat -QfWbigdisplay
```

the flags list would contain

```
qstat -Q -f -W bigdisplay
```

The operands list contains all other command line arguments following the flags. There will always be at least one element in operands because if no operands are typed by the user, the default destination or server name is used. The objects list contains all the information retrieved from the server(s) so the Tcl interpreter can run once to format the entire output. This list has the same number of elements as the operands list. Each element is another list with two elements.

The first element is a string giving the type of objects to be found in the second. The string can take the values "server", "queue", "job" or "error".

The second element will be a list in which each element is a single batch status object of the type given by the string discussed above. In the case of "error", the list will be empty. Each object is again a list. The first element is the name of the object. The second is a list of attributes.

The third element will be the object text.

All three of these object elements correspond with fields in the structure `batch_status` which is described in detail for each type of object by the man pages for `pbs_statjob(3)`, `pbs_statque(3)`, and `pbs_statsserver(3)`. Each attribute in the second element list whose elements correspond with the `attr1` structure. Each will be a list with two elements. The first will be the attribute name and the second will be the attribute value.

## Exit status

Upon successful processing of all the operands presented to the `qstat` command, the exit status will be a value of zero.

If the `qstat` command fails to process any operand, the command exits with a value greater than zero.

### Related topics

- [qalter\(1B\)](#)
- [qsub\(1B\)](#)

### Non-Adaptive Computing topics

- `pbs_alterjob(3B)`
- `pbs_statjob(3B)`
- `pbs_statque(3B)`
- `pbs_statsserver(3B)`
- `pbs_submit(3B)`
- `pbs_job_attributes(7B)`
- `pbs_queue_attributes(7B)`
- `pbs_server_attributes(7B)`
- `qmgr query_other_jobs` parameter (allow non-admin users to see other users' jobs)
- `pbs_resources_*(7B)` where \* is system type
- PBS ERS

## qsub

Submit PBS job.

## Synopsis

```
qsub [-a date_time] [-A account_string] [-b secs] [-c checkpoint_options]
[-C directive_prefix] [-d path] [-D path] [-e path] [-f] [-F] [-h]
[-I ] [-j join ] [-k keep ] [-l resource_list ]
[-m mail_options] [-M user_list] [-n] [-N name] [-o path]
[-p priority] [-P user[:group]] [-q destination] [-r c] [-S path_to_shell(s)]
[-t array_request] [-u user_list]
[-v variable_list] [-V] [-W additional_attributes] [-x] [-X] [-z] [script]
```

## Description

To create a job is to submit an executable script to a batch server. The batch server will be the default server unless the **-q** option is specified. The command parses a script prior to the actual script execution; it does not execute a script itself. All script-writing rules remain in effect, including the **"#!"** at the head of the file (see discussion of **PBS\_DEFAULT** under [Environment variables on page 2760](#)). Typically, the script is a shell script which will be executed by a command shell such as **sh** or **csh**.

Options on the **qsub** command allow the specification of attributes which affect the behavior of the job.

The **qsub** command will pass certain environment variables in the **Variable\_List** attribute of the job. These variables will be available to the job. The value for the following variables will be taken from the environment of the **qsub** command: **HOME**, **LANG**, **LOGNAME**, **PATH**, **MAIL**, **SHELL**, and **TZ**. These values will be assigned to a new name which is the current name prefixed with the string **"PBS\_O\_"**. For example, the job will have access to an environment variable named **PBS\_O\_HOME** which have the value of the variable **HOME** in the **qsub** command environment.

In addition to the above, the following environment variables will be available to the batch job:

Variable	Description
<b>PBS_O_HOST</b>	The name of the host upon which the <b>qsub</b> command is running.
<b>PBS_SERVER</b>	The hostname of the <b>pbs_server</b> which <b>qsub</b> submits the job to.
<b>PBS_O_QUEUE</b>	The name of the original queue to which the job was submitted.
<b>PBS_O_WORKDIR</b>	The absolute path of the current working directory of the <b>qsub</b> command.
<b>PBS_ARRAYID</b>	Each member of a job array is assigned a unique identifier (see <b>-t</b> option).
<b>PBS_ENVIRONMENT</b>	Set to <b>PBS_BATCH</b> to indicate the job is a batch job, or to <b>PBS_INTERACTIVE</b> to indicate the job is a PBS interactive job (see <b>-I</b> option).
<b>PBS_GPUFILE</b>	The name of the file containing the list of assigned GPUs. For more information about how to set up TORQUE with GPUS, see <a href="#">Accelerators on page 815</a> .

Variable	Description
<b>PBS_JOBID</b>	The job identifier assigned to the job by the batch system. It can be used in the stdout and stderr paths. TORQUE replaces <code>\$PBS_JOBID</code> with the job's jobid (for example, <code>#PBS -o /tmp/\$PBS_JOBID.output</code> ).
<b>PBS_JOBNAME</b>	The job name supplied by the user.
<b>PBS_NODEFILE</b>	The name of the file contains the list of nodes assigned to the job (for parallel and cluster systems).
<b>PBS_QUEUE</b>	The name of the queue from which the job is executed.

## Options

Option	Name	Description
<b>-a</b>	date_time	<p>Declares the time after which the job is eligible for execution.</p> <p>The <code>date_time</code> argument is in the form:</p> <pre>[[ [CC] YY] MM] DD] hhmm [ . SS]</pre> <p>where <i>CC</i> is the first two digits of the year (the century), <i>YY</i> is the second two digits of the year, <i>MM</i> is the two digits for the month, <i>DD</i> is the day of the month, <i>hh</i> is the hour, <i>mm</i> is the minute, and the optional <i>SS</i> is the seconds.</p> <p>If the month (<i>MM</i>) is not specified, it will default to the current month if the specified day (<i>DD</i>) is in the future. Otherwise, the month will be set to next month. Likewise, if the day (<i>DD</i>) is not specified, it will default to today if the time (<i>hhmm</i>) is in the future. Otherwise, the day will be set to tomorrow.</p> <p>For example, if you submit a job at 11:15 am with a time of <code>-a 1110</code>, the job will be eligible to run at 11:10 am tomorrow.</p>
<b>-A</b>	account_string	<p>Defines the account string associated with the job. The <code>account_string</code> is an undefined string of characters and is interpreted by the server which executes the job. See section 2.7.1 of the PBS ERS.</p>
<b>-b</b>	seconds	<p>Defines the maximum number of seconds <code>qsub</code> will block attempting to contact <code>pbs_server</code>. If <code>pbs_server</code> is down, or for a variety of communication failures, <code>qsub</code> will continually retry connecting to <code>pbs_server</code> for job submission.</p> <p>This value overrides the <code>CLIENTRETRY</code> parameter in <code>torque.cfg</code>. This is a non-portable TORQUE extension. Portability-minded users can use the <code>PBS_CLIENTRETRY</code> environmental variable. A negative value is interpreted as infinity. The default is 0.</p>

Option	Name	Description
<b>-c</b>	checkpoint_options	<p>Defines the options that will apply to the job. If the job executes upon a host which does not support checkpoint, these options will be ignored.</p> <p>Valid checkpoint options are:</p> <ul style="list-style-type: none"> <li>• <i>none</i> – No checkpointing is to be performed.</li> <li>• <i>enabled</i> – Specify that checkpointing is allowed but must be explicitly invoked by either the <a href="#">ghold</a> or <a href="#">gchkpt</a> commands.</li> <li>• <i>shutdown</i> – Specify that checkpointing is to be done on a job at pbs_mom shutdown.</li> <li>• <i>periodic</i> – Specify that periodic checkpointing is enabled. The default interval is 10 minutes and can be changed by the \$checkpoint_interval option in the MOM config file or by specifying an interval when the job is submitted</li> <li>• <i>interval=minutes</i> – Checkpointing is to be performed at an interval of minutes, which is the integer number of minutes of wall time used by the job. This value must be greater than zero.</li> <li>• <i>depth=number</i> – Specify a number (depth) of checkpoint images to be kept in the checkpoint directory.</li> <li>• <i>dir=path</i> – Specify a checkpoint directory (default is /var/spool/torque/checkpoint).</li> </ul>
<b>-C</b>	directive_prefix	<p>Defines the prefix that declares a directive to the qsub command within the script file. (See the paragraph on script directives under <a href="#">Extended description on page 2761</a>.)</p> <p>If the -C option is presented with a directive_prefix argument that is the null string, qsub will not scan the script file for directives.</p>
<b>-d</b>	path	<p>Defines the working directory path to be used for the job. If the -d option is not specified, the default working directory is the home directory. This option sets the environment variable PBS_O_INITDIR.</p>
<b>-D</b>	path	<p>Defines the root directory to be used for the job. This option sets the environment variable PBS_O_ROOTDIR.</p>

Option	Name	Description
<b>-e</b>	path	<p>Defines the path to be used for the standard error stream of the batch job. The path argument is of the form:</p> <pre>[hostname:]path_name</pre> <p>where <i>hostname</i> is the name of a host to which the file will be returned, and <i>path_name</i> is the path name on that host in the syntax recognized by POSIX.</p> <div data-bbox="516 495 1409 585" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p> When specifying a directory for the location you need to include a trailing slash.</p> </div> <p>The argument will be interpreted as follows:</p> <ul style="list-style-type: none"> <li>• <i>path_name</i> – where <i>path_name</i> is not an absolute path name, then the <code>qsub</code> command will expand the path name relative to the current working directory of the command. The command will supply the name of the host upon which it is executing for the hostname component.</li> <li>• <i>hostname:path_name</i> – where <i>path_name</i> is not an absolute path name, then the <code>qsub</code> command will not expand the path name relative to the current working directory of the command. On delivery of the standard error, the path name will be expanded relative to the user's home directory on the hostname system.</li> <li>• <i>path_name</i> – where <i>path_name</i> specifies an absolute path name, then the <code>qsub</code> will supply the name of the host on which it is executing for the hostname.</li> <li>• <i>hostname:path_name</i> – where <i>path_name</i> specifies an absolute path name, the path will be used as specified.</li> </ul> <p>If the <code>-e</code> option is not specified, the default file name for the standard error stream will be used. The default name has the following form:</p> <ul style="list-style-type: none"> <li>• <i>job_name.esquence_number</i> – where <i>job_name</i> is the name of the job (see the <code>-n</code> name option) and <i>sequence_number</i> is the job number assigned when the job is submitted.</li> </ul>
<b>-f</b>	---	<p>Job is made fault tolerant. Jobs running on multiple nodes are periodically polled by mother superior. If one of the nodes fails to report, the job is canceled by mother superior and a failure is reported. If a job is fault tolerant, it will not be canceled based on failed polling (no matter how many nodes fail to report). This may be desirable if transient network failures are causing large jobs not to complete, where ignoring one failed polling attempt can be corrected at the next polling attempt.</p> <div data-bbox="516 1478 1409 1602" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p> If TORQUE is compiled with <code>PBS_NO_POSIX_VIOLATION</code> (there is no config option for this), you have to use <code>-W fault_tolerant=true</code> to mark the job as fault tolerant.</p> </div>

Option	Name	Description
<b>-F</b>	---	<p>Specifies the arguments that will be passed to the job script when the script is launched. The accepted syntax is:</p> <pre>qsub -F "myarg1 myarg2 myarg3=myarg3value" myscript2.sh</pre> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> Quotation marks are required. <code>qsub</code> will fail with an error message if the argument following <code>-F</code> is not a quoted value. The <code>pbs_mom</code> server will pass the quoted value as arguments to the job script when it launches the script.</p> </div>
<b>-h</b>	---	Specifies that a user hold be applied to the job at submission time.
<b>-i</b>	---	<p>Declares that the job is to be run "interactively". The job will be queued and scheduled as any PBS batch job, but when executed, the standard input, output, and error streams of the job are connected through <code>qsub</code> to the terminal session in which <code>qsub</code> is running. Interactive jobs are forced to not rerunable. See <a href="#">Extended description on page 2761</a> for additional information of interactive jobs.</p>
<b>-j</b>	join	<p>Declares if the standard error stream of the job will be merged with the standard output stream of the job.</p> <p>An option argument value of <code>oe</code> directs that the two streams will be merged, intermixed, as standard output. An option argument value of <code>eo</code> directs that the two streams will be merged, intermixed, as standard error.</p> <p>If the join argument is <code>n</code> or the option is not specified, the two streams will be two separate files.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p><b>i</b> If using either the <code>-e</code> or the <code>-o</code> option and the <code>-j eo oe</code> option, the <code>-j</code> option takes precedence and all standard error and output messages go to the chosen output file.</p> </div>

Option	Name	Description
<b>-k</b>	keep	<p>Defines which (if either) of standard output or standard error will be retained on the execution host. If set for a stream, this option overrides the path name for that stream. If not set, neither stream is retained on the execution host.</p> <p>The argument is either the single letter "e" or "o", or the letters "e" and "o" combined in either order. Or the argument is the letter "n".</p> <ul style="list-style-type: none"> <li>• <i>e</i> – The standard error stream is to be retained on the execution host. The stream will be placed in the home directory of the user under whose user id the job executed. The file name will be the default file name given by: <code>job_name.esquence</code> where <i>job_name</i> is the name specified for the job, and <i>sequence</i> is the sequence number component of the job identifier.</li> <li>• <i>o</i> – The standard output stream is to be retained on the execution host. The stream will be placed in the home directory of the user under whose user id the job executed. The file name will be the default file name given by: <code>job_name.osequence</code> where <i>job_name</i> is the name specified for the job, and <i>sequence</i> is the sequence number component of the job identifier.</li> <li>• <i>eo</i> – Both the standard output and standard error streams will be retained.</li> <li>• <i>oe</i> – Both the standard output and standard error streams will be retained.</li> <li>• <i>n</i> – Neither stream is retained.</li> </ul>
<b>-l</b>	resource_list	<p>Defines the resources that are required by the job and establishes a limit to the amount of resource that can be consumed. If not set for a generally available resource, such as CPU time, the limit is infinite. The <b>resource_list</b> argument is of the form: <code>resource_name [= [value]] [, resource_name [= [value]] , ...]</code></p> <div data-bbox="516 1199 1409 1291" style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> In this situation, you should request the more inclusive resource first. For example, a request for procs should come before a gres request.</p> </div> <p>In TORQUE 3.0.2 or later, <code>qsub</code> supports the mapping of <code>-l gpus=X to -l gres=gpus:X</code>. This allows users who are using NUMA systems to make requests such as <code>-l ncpus=20:gpus=5</code> indicating they are not concerned with the GPUs in relation to the NUMA nodes they request, they only want a total of 20 cores and 5 GPUs. For more information, see <a href="#">Requesting resources on page 2586</a>.</p> <p>For information on specifying multiple types of resources for allocation, see .</p>

Option	Name	Description
<b>-m</b>	mail_options	<p>Defines the set of conditions under which the execution server will send a mail message about the job. The mail_options argument is a string which consists of either the single character "n", or one or more of the characters "a", "b", and "e".</p> <p>If the character "n" is specified, no normal mail is sent. Mail for job cancels and other events outside of normal job processing are still sent.</p> <p>For the letters "a", "b", and "e":</p> <ul style="list-style-type: none"> <li>• <i>a</i> – Mail is sent when the job is aborted by the batch system.</li> <li>• <i>b</i> – Mail is sent when the job begins execution.</li> <li>• <i>e</i> – Mail is sent when the job terminates.</li> </ul> <p>If the <code>-m</code> option is not specified, mail will be sent if the job is aborted.</p>
<b>-M</b>	user_list	<p>Declares the list of users to whom mail is sent by the execution server when it sends mail about the job.</p> <p>The user_list argument is of the form:</p> <pre>user[@host] [, user[@host] , ... ]</pre> <p>If unset, the list defaults to the submitting user at the qsub host, i.e. the job owner.</p>
<b>-n</b>	node-exclusive	<p>Allows a user to specify an exclusive-node access/allocation request for the job. This affects only cpusets and compatible schedulers (see <a href="#">Linux cpuset support on page 2623</a>).</p>
<b>-N</b>	name	<p>Declares a name for the job. The name specified may be an unlimited number of characters in length. It must consist of printable, nonwhite space characters with the first character alphabetic.</p> <p>If the <code>-N</code> option is not specified, the job name will be the base name of the job script file specified on the command line. If no script file name was specified and the script was read from the standard input, then the job name will be set to STDIN.</p>

Option	Name	Description
<b>-o</b>	path	<p>Defines the path to be used for the standard output stream of the batch job. The path argument is of the form:</p> <pre>[hostname:]path_name</pre> <p>where <i>hostname</i> is the name of a host to which the file will be returned, and <i>path_name</i> is the path name on that host in the syntax recognized by POSIX.</p> <div style="border: 1px solid black; border-radius: 10px; padding: 5px; margin: 10px 0;">  When specifying a directory for the location you need to include a trailing slash. </div> <p>The argument will be interpreted as follows:</p> <ul style="list-style-type: none"> <li>• <i>path_name</i> – where <i>path_name</i> is not an absolute path name, then the <code>qsub</code> command will expand the path name relative to the current working directory of the command. The command will supply the name of the host upon which it is executing for the hostname component.</li> <li>• <i>hostname:path_name</i> – where <i>path_name</i> is not an absolute path name, then the <code>qsub</code> command will not expand the path name relative to the current working directory of the command. On delivery of the standard output, the path name will be expanded relative to the user's home directory on the hostname system.</li> <li>• <i>path_name</i> – where <i>path_name</i> specifies an absolute path name, then the <code>qsub</code> will supply the name of the host on which it is executing for the hostname.</li> <li>• <i>hostname:path_name</i> where <i>path_name</i> specifies an absolute path name, the path will be used as specified.</li> </ul> <p>If the <code>-o</code> option is not specified, the default file name for the standard output stream will be used. The default name has the following form:</p> <ul style="list-style-type: none"> <li>• <i>job_name.osequence_number</i> – where <i>job_name</i> is the name of the job (see the <a href="#">-n</a> name option) and <i>sequence_number</i> is the job number assigned when the job is submitted.</li> </ul>
<b>-p</b>	priority	<p>Defines the priority of the job. The priority argument must be a integer between -1024 and +1023 inclusive. The default is no priority which is equivalent to a priority of zero.</p>
<b>-P</b>	user [:group]	<p>Allows a root user or manager to submit a job as another user. TORQUE treats proxy jobs as though the jobs were submitted by the supplied username. This feature is available in TORQUE 2.4.7 and later, however, TORQUE 2.4.7 does not have the ability to supply the <code>[:group]</code> option; it is available in TORQUE 2.4.8 and later.</p>

Option	Name	Description
<b>-q</b>	destination	<p>Defines the destination of the job. The destination names a queue, a server, or a queue at a server.</p> <p>The <code>qsub</code> command will submit the script to the server defined by the destination argument. If the destination is a routing queue, the job may be routed by the server to a new destination.</p> <p>If the <code>-q</code> option is not specified, the <code>qsub</code> command will submit the script to the default server. (See <a href="#">Environment variables on page 2760</a> and the PBS ERS section 2.7.4, "Default Server".)</p> <p>If the <code>-q</code> option is specified, it is in one of the following three forms:</p> <ul style="list-style-type: none"> <li>• queue</li> <li>• @server</li> <li>• queue@server</li> </ul> <p>If the destination argument names a queue and does not name a server, the job will be submitted to the named queue at the default server.</p> <p>If the destination argument names a server and does not name a queue, the job will be submitted to the default queue at the named server.</p> <p>If the destination argument names both a queue and a server, the job will be submitted to the named queue at the named server.</p>
<b>-r</b>	y/n	<p>Declares whether the job is rerunnable (see the <a href="#">qrerun</a> command). The option argument is a single character, either y or n.</p> <p>If the argument is "y", the job is rerunnable. If the argument is "n", the job is not rerunnable. The default value is y, rerunnable.</p>
<b>-S</b>	path_list	<p>Declares the path to the desired shell for this job.</p> <pre>qsub script.sh -S /bin/tcsh</pre> <p>If the shell path is different on different compute nodes, use the following syntax:</p> <pre>path[@host] [,path[@host], ...]</pre> <pre>qsub script.sh -S /bin/tcsh@node1,/usr/bin/tcsh@node2</pre> <p>Only one path may be specified for any host named. Only one path may be specified without the corresponding host name. The path selected will be the one with the host name that matched the name of the execution host. If no matching host is found, then the path specified without a host will be selected, if present.</p> <p>If the <code>-S</code> option is not specified, the option argument is the null string, or no entry from the <code>path_list</code> is selected, the execution will use the user's login shell on the execution host.</p>

Option	Name	Description
<b>-t</b>	array_request	<p>Specifies the task ids of a job array. Single task arrays are allowed.</p> <p>The <code>array_request</code> argument is an integer id or a range of integers. Multiple ids or id ranges can be combined in a comma delimited list. Examples: <code>-t 1-100</code> or <code>-t 1,10,50-100</code></p> <p>An optional <i>slot limit</i> can be specified to limit the amount of jobs that can run concurrently in the job array. The default value is unlimited. The slot limit must be the last thing specified in the <code>array_request</code> and is delimited from the array by a percent sign (%).</p> <pre>qsub script.sh -t 0-299%5</pre> <p>This sets the slot limit to 5. Only 5 jobs from this array can run at the same time.</p> <p>You can use <a href="#">qalter</a> to modify slot limits on an array. The server parameter <a href="#">max_slot_limit</a> can be used to set a global slot limit policy.</p>
<b>-u</b>	user_list	<p>Defines the user name under which the job is to run on the execution system.</p> <p>The <code>user_list</code> argument is of the form:</p> <pre>user[@host] [,user[@host],...]</pre> <p>Only one user name may be given per specified host. Only one of the user specifications may be supplied without the corresponding host specification. That user name will be used for execution on any host not named in the argument list. If unset, the user list defaults to the user who is running <code>qsub</code>.</p>
<b>-v</b>	variable_list	<p>Expands the list of environment variables that are exported to the job.</p> <p>In addition to the variables described in the "Description" section above, <code>variable_list</code> names environment variables from the <code>qsub</code> command environment which are made available to the job when it executes. The <code>variable_list</code> is a comma separated list of strings of the form <code>variable</code> or <code>variable=value</code>. These variables and their values are passed to the job. Note that <code>-v</code> has a higher precedence than <code>-V</code>, so identically named variables specified via <code>-v</code> will provide the final value for an environment variable in the job.</p>
<b>-V</b>	---	<p>Declares that all environment variables in the <code>qsub</code> commands environment are to be exported to the batch job.</p>

Option	Name	Description
-W	additional_attributes	<p>The <code>-W</code> option allows for the specification of additional job attributes. The general syntax of <code>-W</code> is in the form:</p> <pre>-W attr_name=attr_value.</pre> <p>You can use multiple <code>-W</code> options with this syntax:</p> <pre>-W attr_name1=attr_value1 -W attr_name2=attr_value2.</pre> <div data-bbox="544 499 1432 625" style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p><b>i</b> If white space occurs anywhere within the option argument string or the equal sign, "=", occurs within an attribute_value string, then the string must be enclosed with either single or double quote marks.</p> </div> <p>PBS currently supports the following attributes within the <code>-W</code> option:</p> <ul style="list-style-type: none"> <li>• <code>depend=dependency_list</code> – Defines the dependency between this and other jobs. The <code>dependency_list</code> is in the form: <pre>type[:argument[:argument...]][,type:argument...]</pre> <p>The argument is either a numeric count or a PBS job id according to type. If argument is a count, it must be greater than 0. If it is a job id and not fully specified in the form <code>seq_number.server.name</code>, it will be expanded according to the default server rules which apply to job IDs on most commands. If argument is null (the preceding colon need not be specified), the dependency of the corresponding type is cleared (unset). For more information, see <a href="#">depend=dependency_list valid dependencies on page 2756</a>.</p> </li> <li>• <code>group_list=g_list</code> – Defines the group name under which the job is to run on the execution system. The <code>g_list</code> argument is of the form: <pre>group[@host][,group[@host],...]</pre> <p>Only one group name may be given per specified host. Only one of the group specifications may be supplied without the corresponding host specification. That group name will used for execution on any host not named in the argument list. If not set, the <code>group_list</code> defaults to the primary group of the user under which the job will be run.</p> </li> <li>• <code>interactive=true</code> – If the interactive attribute is specified, the job is an interactive job. The <code>-I</code> option is an alternative method of specifying this attribute.</li> <li>• <code>job_radix=&lt;int&gt;</code> – To be used with parallel jobs. It directs the Mother Superior of the job to create a distribution radix of size <code>&lt;int&gt;</code> between sisters. See <a href="#">Managing multi-node jobs on page 2585</a>.</li> <li>• <code>stagein=file_list</code></li> <li>• <code>stageout=file_list</code> – Specifies which files are staged (copied) in before job start or staged out after the job completes execution. On completion of the job, all staged-in and staged-out files are removed from the execution system. The <code>file_list</code> is in the form: <pre>local_file@hostname:remote_file[,...]</pre> <p>regardless of the direction of the copy. The name <code>local_file</code> is the name of the file on the system where the job executed. It may be an absolute path or relative to the home directory of the user. The name <code>remote_file</code> is the destination name on the host specified by <code>hostname</code>. The name may be</p> </li> </ul>

Option	Name	Description
		<p>absolute or relative to the user's home directory on the destination host. The use of wildcards in the file name is not recommended. The file names map to a remote copy program (rcp) call on the execution system in the follow manner:</p> <ul style="list-style-type: none"> <li>◦ For stagein: <code>rcp hostname:remote_file local_file</code></li> <li>◦ For stageout: <code>rcp local_file hostname:remote_file</code></li> </ul> <p>Data staging examples:</p> <pre>-W stagein=/tmp/input.txt@headnode:/home/user/input.txt -W stageout=/tmp/output.txt@headnode:/home/user/output.txt</pre> <p>If TORQUE has been compiled with wordexp support, then variables can be used in the specified paths. Currently only \$PBS_JOBID, \$HOME, and \$TMPDIR are supported for stagein.</p> <ul style="list-style-type: none"> <li>• <code>umask=XXX</code> – Sets umask used to create stdout and stderr spool files in pbs_mom spool directory. Values starting with 0 are treated as octal values, otherwise the value is treated as a decimal umask value.</li> </ul>
<b>-x</b>	---	<p>By default, if you submit an interactive job with a script, the script will be parsed for PBS directives but the rest of the script will be ignored since it's an interactive job. The <code>-x</code> option allows the script to be executed in the interactive job and then the job completes. For example:</p> <pre>script.sh #!/bin/bash ls ---end script---</pre> <pre>qsub -I script.sh qsub: waiting for job 5.napali to start dbeer@napali:~# &lt;displays the contents of the directory, because of the ls command&gt; qsub: job 5.napali completed</pre>
<b>-X</b>	---	Enables X11 forwarding. The DISPLAY environment variable must be set.
<b>-z</b>	---	Directs that the <code>qsub</code> command is not to write the job identifier assigned to the job to the commands standard output.

### depend=dependency\_list valid dependencies



For job dependencies to work correctly, you must set the [keep\\_completed](#) on page 2772 server parameter.

Dependency	Description
<code>synccount:count</code>	This job is the first in a set of jobs to be executed at the same time. Count is the number of additional jobs in the set.
<code>syncwith:jobid</code>	This job is an additional member of a set of jobs to be executed at the same time. In the above and following dependency types, jobid is the job identifier of the first job in the set.
<code>after:jobid[:jobid...]</code>	This job may be scheduled for execution at any point after jobs jobid have started execution.
<code>afterok:jobid[:jobid...]</code>	This job may be scheduled for execution only after jobs jobid have terminated with no errors. See the csh warning under <a href="#">Extended description on page 2761</a> .
<code>afternotok:jobid[:jobid...]</code>	This job may be scheduled for execution only after jobs jobid have terminated with errors. See the csh warning under <a href="#">Extended description on page 2761</a> .
<code>afterany:jobid[:jobid...]</code>	This job may be scheduled for execution after jobs jobid have terminated, with or without errors.
<code>on:count</code>	This job may be scheduled for execution after count dependencies on other jobs have been satisfied. This form is used in conjunction with one of the "before" forms (see below).
<code>before:jobid[:jobid...]</code>	When this job has begun execution, then jobs jobid... may begin.
<code>beforeok:jobid[:jobid...]</code>	If this job terminates execution without errors, then jobs jobid... may begin. See the csh warning under <a href="#">Extended description on page 2761</a> .
<code>beforenotok:jobid[:jobid...]</code>	If this job terminates execution with errors, then jobs jobid... may begin. See the csh warning under <a href="#">Extended description on page 2761</a> .

Dependency	Description
beforeany:jobid[:jobid...]	<p>When this job terminates execution, jobs jobid... may begin.</p> <p>If any of the before forms are used, the jobs referenced by jobid must have been submitted with a dependency type of on.</p> <p>If any of the before forms are used, the jobs referenced by jobid must have the same owner as the job being submitted. Otherwise, the dependency is ignored.</p>
<p><b>i</b> Array dependencies make a job depend on an array or part of an array. If no count is given, then the entire array is assumed. For examples, see <a href="#">Dependency examples on page 2759</a>.</p>	
afterstartarray:arrayid[count]	<p>After this many jobs have started from arrayid, this job may start.</p>
afterokarray:arrayid[count]	<p>This job may be scheduled for execution only after jobs in arrayid have terminated with no errors.</p>
afternotokarray:arrayid[count]	<p>This job may be scheduled for execution only after jobs in arrayid have terminated with errors.</p>
afteranyarray:arrayid[count]	<p>This job may be scheduled for execution after jobs in arrayid have terminated, with or without errors.</p>
beforestartarray:arrayid[count]	<p>Before this many jobs have started from arrayid, this job may start.</p>
beforeokarray:arrayid[count]	<p>If this job terminates execution without errors, then jobs in arrayid may begin.</p>
beforenotokarray:arrayid[count]	<p>If this job terminates execution with errors, then jobs in arrayid may begin.</p>
beforeanyarray:arrayid[count]	<p>When this job terminates execution, jobs in arrayid may begin.</p> <p>If any of the before forms are used, the jobs referenced by arrayid must have been submitted with a dependency type of on.</p> <p>If any of the before forms are used, the jobs referenced by arrayid must have the same owner as the job being submitted. Otherwise, the dependency is ignored.</p>

Dependency	Description
<p><b>i</b> Error processing of the existence, state, or condition of the job on which the newly submitted job is a deferred service, i.e. the check is performed after the job is queued. If an error is detected, the new job will be deleted by the server. Mail will be sent to the job submitter stating the error.</p>	

## Dependency examples

```
qsub -W depend=afterok:123.big.iron.com /tmp/script
```

```
qsub -W depend=before:234.hunk1.com:235.hunk1.com
```

```
/tmp/script
```

```
qsub script.sh -W depend=afterokarray:427 []
```

(This assumes every job in array 427 has to finish successfully for the dependency to be satisfied.)

```
qsub script.sh -W depend=afterokarray:427 [] [5]
```

(This means that 5 of the jobs in array 427 have to successfully finish in order for the dependency to be satisfied.)

## Operands

The `qsub` command accepts a script operand that is the path to the script of the job. If the path is relative, it will be expanded relative to the working directory of the `qsub` command.

If the script operand is not provided or the operand is the single character "-", the `qsub` command reads the script from standard input. When the script is being read from Standard Input, `qsub` will copy the file to a temporary file. This temporary file is passed to the library interface routine `pbs_submit`. The temporary file is removed by `qsub` after `pbs_submit` returns or upon the receipt of a signal which would cause `qsub` to terminate.

## Standard input

The `qsub` command reads the script for the job from standard input if the script operand is missing or is the single character "-".

## Input files

The script file is read by the `qsub` command. `qsub` acts upon any directives found in the script.

When the job is created, a copy of the script file is made and that copy cannot be modified.

## Standard output

Unless the `-z` option is set, the job identifier assigned to the job will be written to standard output if the job is successfully created.

## Standard error

The `qsub` command will write a diagnostic message to standard error for each error occurrence.

## Environment variables

The values of some or all of the variables in the `qsub` commands environment are exported with the job (see the `-v` and `-V` options).

The environment variable `PBS_DEFAULT` defines the name of the default server. Typically, it corresponds to the system name of the host on which the server is running. If `PBS_DEFAULT` is not set, the default is defined by an administrator established file.

The environment variable `PBS_DPREFIX` determines the prefix string which identifies directives in the script.

The environment variable `PBS_CLIENTRETRY` defines the maximum number of seconds `qsub` will block (see the `-b` option). Despite the name, currently `qsub` is the only client that supports this option.

## torque.cfg

The `torque.cfg` file, located in `PBS_SERVER_HOME` (`/var/spool/torque` by default) controls the behavior of the `qsub` command. This file contains a list of parameters and values separated by whitespace.

- *QSUBSLEEP* – takes an integer operand which specifies time to sleep when running `qsub` command. Used to prevent users from overwhelming the scheduler.
- *SUBMITFILTER* – specifies the path to the submit filter used to pre-process job submission. The default path is `libexecdir/qsub_filter`, which falls back to `/usr/local/sbin/torque_submitfilter` for backwards compatibility. This `torque.cfg` parameter overrides this default.
- *SERVERHOST*
- *QSUBHOST*
- *QSUBSENDUID*
- *XAUTHPATH*
- *CLIENTRETRY*
- *VALIDATEGROUP*
- *DEFAULTCKPT*
- *VALIDATEPATH*
- *RERUNNABLEBYDEFAULT*

For example:

```
QSUBSLEEP 2
```

```
RERUNNABLEBYDEFAULT false
```

## Extended description

### Script Processing:

A job script may consist of PBS directives, comments and executable statements. A PBS directive provides a way of specifying job attributes in addition to the command line options. For example:

```

:
#PBS -N Job_name
#PBS -l walltime=10:30,mem=320kb
#PBS -m be
#
step1 arg1 arg2
step2 arg3 arg4

```

The `qsub` command scans the lines of the script file for directives. An initial line in the script that begins with the characters `#!/` or the character `:` will be ignored and scanning will start with the next line. Scanning will continue until the first executable line, that is a line that is not blank, not a directive line, nor a line whose first nonwhite space character is `#`. If directives occur on subsequent lines, they will be ignored.

A line in the script file will be processed as a directive to `qsub` if and only if the string of characters starting with the first nonwhite space character on the line and of the same length as the directive prefix matches the directive prefix.

The remainder of the directive line consists of the options to `qsub` in the same syntax as they appear on the command line. The option character is to be preceded with the `-` character.

If an option is present in both a directive and on the command line, that option and its argument, if any, will be ignored in the directive. The command line takes precedence.

If an option is present in a directive and not on the command line, that option and its argument, if any, will be processed as if it had occurred on the command line.

The directive prefix string will be determined in order of preference from:

- The value of the `-c` option argument if the option is specified on the command line.
- The value of the environment variable `PBS_DPREFIX` if it is defined.
- The four character string `#PBS`.

If the `-c` option is found in a directive in the script file, it will be ignored.

### User Authorization:

When the user submits a job from a system other than the one on which the PBS Server is running, the name under which the job is to be executed is selected according to the rules listed under the `-u` option. The user submitting the job must be authorized to run the job under the execution user name. This authorization is provided if:

- The host on which `qsub` is run is trusted by the execution host (see `/etc/hosts.equiv`).
- The execution user has an `.rhosts` file naming the submitting user on the submitting host.

## C-Shell .logout File:

The following warning applies for users of the c-shell, csh. If the job is executed under the csh and a `.logout` file exists in the home directory in which the job executes, the exit status of the job is that of the `.logout` script, not the job script. This may impact any inter-job dependencies. To preserve the job exit status, either remove the `.logout` file or place the following line as the first line in the `.logout` file:

```
set EXITVAL = $status
```

and the following line as the last executable line in `.logout`:

```
exit $EXITVAL
```

## Interactive Jobs:

If the `-i` option is specified on the command line or in a script directive, or if the "interactive" job attribute declared true via the `-W` option, `-W interactive=true`, either on the command line or in a script directive, the job is an interactive job. The script will be processed for directives, but will not be included with the job. When the job begins execution, all input to the job is from the terminal session in which `qsub` is running.

When an interactive job is submitted, the `qsub` command will not terminate when the job is submitted. `qsub` will remain running until the job terminates, is aborted, or the user interrupts `qsub` with a SIGINT (the control-C key). If `qsub` is interrupted prior to job start, it will query if the user wishes to exit. If the user response "yes", `qsub` exits and the job is aborted.

Once the interactive job has started execution, input to and output from the job pass through `qsub`. Keyboard generated interrupts are passed to the job. Lines entered that begin with the tilde (~) character and contain special sequences are escaped by `qsub`. The recognized escape sequences are:

Sequence	Description
~.	<code>qsub</code> terminates execution. The batch job is also terminated.
~susp	Suspend the <code>qsub</code> program if running under the C shell. "susp" is the suspend character (usually CNTL-Z).
~asusp	Suspend the input half of <code>qsub</code> (terminal to job), but allow output to continue to be displayed. Only works under the C shell. "asusp" is the auxiliary suspend character, usually CNTL-Y.

## Exit status

Upon successful processing, the `qsub` exit status will be a value of zero.

If the `qsub` command fails, the command exits with a value greater than zero.

### Related topics

- [galter\(1B\)](#)
- [qdel\(1B\)](#)

- [qhold](#)(1B)
- [qrls](#)(1B)
- [qsig](#)(1B)
- [qstat](#)(1B)
- [pbs\\_server](#)(8B)

### Non-Adaptive Computing topics

- [pbs\\_connect](#)(3B)
- [pbs\\_job\\_attributes](#)(7B)
- [pbs\\_queue\\_attributes](#)(7B)
- [pbs\\_resources\\_iris5](#)(7B)
- [pbs\\_resources\\_sp2](#)(7B)
- [pbs\\_resources\\_sunos4](#)(7B)
- [pbs\\_resources\\_unicos8](#)(7B)
- [pbs\\_server\\_attributes](#)(7B)
- [qselect](#)(1B)
- [qmove](#)(1B)
- [qmsg](#)(1B)
- [qrerun](#)(1B)

## qterm

Terminate processing by a PBS batch server.

### Synopsis

```
qterm [-t type] [server...]
```

### Description

The `qterm` command terminates a batch server. When a server receives a terminate command, the server will go into the "Terminating" state. No new jobs will be allowed to be started into execution or enqueued into the server. The impact on jobs currently being run by the server depends

In order to execute `qterm`, the user must have PBS Operation or Manager privileges.

## Options

Option	Name	Description
<b>-t</b>	type	<p>Specifies the type of shut down. The types are:</p> <ul style="list-style-type: none"> <li>• <i>immediate</i> – If checkpointing is supported, all running jobs are to immediately stop execution. If checkpointing is supported, running jobs that can be checkpointed are checkpointed, terminated, and requeued. If checkpoint is not supported or the job cannot be checkpointed, running jobs are requeued if the rerunable attribute is true. Otherwise, jobs are killed.</li> <li>• <i>delay</i> – If checkpointing is supported, running jobs that can be checkpointed are checkpointed, terminated, and requeued. If a job cannot be checkpointed, but can be rerun, the job is terminated and requeued. Otherwise, running jobs are allowed to continue to run.</li> </ul> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin: 10px 0;"> <p> Note, the operator or administrator may use the <a href="#">qrerun</a> and <a href="#">qdel</a> commands to remove running jobs.</p> </div> <ul style="list-style-type: none"> <li>• <i>quick</i> – This is the default action if the <code>-t</code> option is not specified. This option is used when you wish that running jobs be left running when the server shuts down. The server will cleanly shutdown and can be restarted when desired. Upon restart of the server, jobs that continue to run are shown as running; jobs that terminated during the server's absence will be placed into the exiting state.</li> </ul>

## Operands

The server operand specifies which servers are to shut down. If no servers are given, then the default server will be terminated.

## Standard error

The `qterm` command will write a diagnostic message to standard error for each error occurrence.

## Exit status

Upon successful processing of all the operands presented to the `qterm` command, the exit status will be a value of zero.

If the `qterm` command fails to process any operand, the command exits with a value greater than zero.

Related topics(non-Adaptive Computing topics)

- [pbs\\_server\(8B\)](#)
- [qmgr\(8B\)](#)
- [pbs\\_resources\\_aix4\(7B\)](#)
- [pbs\\_resources\\_iris5\(7B\)](#)
- [pbs\\_resources\\_sp2\(7B\)](#)

- pbs\_resources\_sunos4(7B)
- pbs\_resources\_unicos8(7B)

## trqauthd

(TORQUE authorization daemon)

### Synopsis

```
trqauthd -D
trqauthd -d
```

### Description

The `trqauthd` daemon, introduced in TORQUE 4.0.0, replaced the `pbs_iff` authentication process. When users connect to `pbs_server` by calling one of the TORQUE utilities or by using the TORQUE APIs, the new user connection must be authorized by a trusted entity which runs as root. The advantage of `trqauthd`'s doing this rather than `pbs_iff` is that `trqauthd` is resident, meaning you do not need to be loaded every time a connection is made; multi-threaded; scalable; and more easily adapted to new functionality than `pbs_iff`.

Beginning in TORQUE 4.2.6, `trqauthd` can remember the currently active `pbs_server` host, enhancing high availability functionality. Previously, `trqauthd` tried to connect to each host in the `$TORQUE_HOME/<server_name>` file until it could successfully connect. Because it now remembers the active server, it tries to connect to that server first. If it fails to connect, it will go through the `<server_name>` file and try to connect to a host where an active `pbs_server` is running.

### Options

-D — Debug	
<b>Format</b>	---
<b>Default</b>	---
<b>Description</b>	Run <code>trqauthd</code> in debug mode.
<b>Example</b>	<code>trqauthd -D</code>

-d — Terminate	
<b>Format</b>	---

<b>-d — Terminate</b>	
<b>Default</b>	---
<b>Description</b>	Terminate trqauthd.
<b>Example</b>	<code>trqauthd -d</code>

## Appendix B: Server parameters

TORQUE server parameters are specified using the [qmgr](#) command. The `set` subcommand is used to modify the **server** object. For example:

```
> qmgr -c 'set server default_queue=batch'
```

### Parameters

<b>acl_hosts</b>	
<b>Format</b>	<HOST>[,<HOST>]... <i>or</i> <HOST>[range] <i>or</i> <HOST*> where the asterisk (*) can appear anywhere in the host name
<b>Default</b>	(Only the host running <code>pbs_server</code> may submit jobs.)
<b>Description</b>	<p>Specifies a list of hosts from which jobs may be submitted. Hosts in the server nodes file located at <code>\$TORQUE/server_priv/nodes</code> cannot be added to the list using the <code>acl_hosts</code> parameter (see <a href="#">Server node file configuration on page 2572</a>). To submit batch or interactive jobs (see <a href="#">Server configuration on page 2566</a>) through hosts that are specified in the server nodes file, use the <a href="#">submit_hosts</a> parameter.</p> <pre>Qmgr: set queue batch acl_hosts = "hostA,hostB" Qmgr: set queue batch acl_hosts += "hostE,hostF,hostG"</pre> <p>In version 2.5 and later, the wildcard (*) character can appear anywhere in the host name, and ranges are supported; these specifications also work for managers and operators.</p> <pre>Qmgr: set server acl_hosts = "galaxy*.tom.org" Qmgr: set server acl_hosts += "galaxy[0-50].tom.org" Qmgr: set server managers+=tom@galaxy[0-50].tom.org</pre>

acl_host_enable	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <b>TRUE</b> , specifies that the <a href="#">acl_hosts</a> value is enabled.

acl_logic_or	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <b>TRUE</b> , the user and group queue ACL's are logically OR'd. When set to <b>FALSE</b> , they are AND'd.

allow_node_submit	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <b>TRUE</b> , specifies that users can submit jobs directly from any trusted compute host directly or from within batch jobs (see <a href="#">Configuring job submission hosts on page 2566</a> ). <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;">  When you enable <b>allow_node_submit</b>, you must also enable the <a href="#">allow_proxy_user on page 2767</a> parameter to allow user proxying when submitting and running jobs. </div>

allow_proxy_user	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <b>TRUE</b> , specifies that users can proxy from one user to another. Proxy requests will be either validated by <code>ruserok()</code> or by the scheduler (see <a href="#">Job submission on page 2583</a> ).

auto_node_np	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>DISABLED</b>
<b>Description</b>	When set to <code>TRUE</code> , automatically configures a node's np (number of processors) value based on the ncpus value from the status update. Requires full manager privilege to set or alter.

automatic_requeue_exit_code	
<b>Format</b>	<LONG>
<b>Default</b>	---
<b>Description</b>	This is an exit code, defined by the admin, that tells pbs_server to requeue the job instead of considering it as completed. This allows the user to add some additional checks that the job can run meaningfully, and if not, then the job script exits with the specified code to be requeued.

checkpoint_defaults	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies for a queue the default checkpoint values for a job that does not have checkpointing specified. The checkpoint_defaults parameter only takes effect on execution queues. <pre>set queue batch checkpoint_defaults="enabled, periodic, interval=5"</pre>

clone_batch_delay	
<b>Format</b>	<INTEGER>
<b>Default</b>	1
<b>Description</b>	Specifies the delay (in seconds) between clone batches (see <a href="#">clone_batch_size</a> ).

## clone\_batch\_size

<b>Format</b>	<INTEGER>
<b>Default</b>	256
<b>Description</b>	Job arrays are created in batches of size <i>X</i> . <i>X</i> jobs are created, and after the <a href="#">clone_batch_delay</a> , <i>X</i> more are created. This repeats until all are created.

## cray\_enabled

<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <code>TRUE</code> , specifies that this instance of <code>pbs_server</code> has Cray hardware that reports to it. See <a href="#">Installation Notes for Moab and TORQUE for Cray on page 1236</a> .

## default\_queue

<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Indicates the queue to assign to a job if no queue is explicitly specified by the submitter.

## disable\_server\_id\_check

<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	<p>When set to <code>TRUE</code>, makes it so the user for the job doesn't have to exist on the server. The user must still exist on all the compute nodes or the job will fail when it tries to execute.</p> <div style="border: 1px solid red; padding: 5px; margin-top: 10px;">  If you have <code>disable_server_id_check</code> set to <code>TRUE</code>, a user could request a group to which they do not belong. Setting <code>VALIDATEGROUP</code> to <code>TRUE</code> in the <code>torque.cfg</code> file prevents such a scenario (see <a href="#">Appendix K: "torque.cfg" configuration file on page 2828</a>). </div>

display_job_server_suffix	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>TRUE</b>
<b>Description</b>	<p>When set to <b>TRUE</b>, TORQUE will display both the job ID and the host name. When set to <b>FALSE</b>, only the job ID will be displayed.</p> <div style="border: 1px solid #0070c0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> If set to <b>FALSE</b>, the environment variable <code>NO_SERVER_SUFFIX</code> must be set to <b>TRUE</b> for <code>pbs_track</code> to work as expected.</p> </div>

interactive_jobs_can_roam	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	<p>By default, interactive jobs run from the login node that they submitted from. When <b>TRUE</b>, interactive jobs may run on login nodes other than the one where the jobs were submitted to. See <a href="#">Installation Notes for Moab and TORQUE for Cray on page 1236</a>.</p>

job_force_cancel_time	
<b>Format</b>	<INTEGER>
<b>Default</b>	Disabled
<b>Description</b>	<p>If a job has been deleted and is still in the system after <i>x</i> seconds, the job will be purged from the system. This is mostly useful when a job is running on a large number of nodes and one node goes down. The job cannot be deleted because the MOM cannot be contacted. The <code>qdel</code> fails and none of the other nodes can be reused. This parameter can be used to remedy such situations.</p>

job_log_file_max_size	
<b>Format</b>	<INTEGER>
<b>Default</b>	---

**job\_log\_file\_max\_size**

<b>Description</b>	This specifies a soft limit (in kilobytes) for the job log's maximum size. The file size is checked every five minutes and if the <i>current day</i> file size is greater than or equal to this value, it is rolled from <filename> to <filename.1> and a new empty log is opened. If the current day file size exceeds the maximum size a second time, the <filename.1> log file is rolled to <filename.2>, the current log is rolled to <filename.1>, and a new empty log is opened. Each new log causes all other logs to roll to an extension that is one greater than its current number. Any value less than 0 is ignored by <code>pbs_server</code> (meaning the log will not be rolled).
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**job\_log\_file\_roll\_depth**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Default</b>	---
----------------	-----

<b>Description</b>	This sets the maximum number of new log files that are kept in a day if the <a href="#">job_log_file_max_size</a> parameter is set. For example, if the roll depth is set to 3, no file can roll higher than <filename.3>. If a file is already at the specified depth, such as <filename.3>, the file is deleted so it can be replaced by the incoming file roll, <filename.2>.
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**job\_log\_keep\_days**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Default</b>	---
----------------	-----

<b>Description</b>	This maintains logs for the number of days designated. If set to 4, any log file older than 4 days old is deleted.
--------------------	--------------------------------------------------------------------------------------------------------------------

**job\_nanny**

<b>Format</b>	<BOOLEAN>
---------------	-----------

<b>Default</b>	<b>FALSE</b>
----------------	--------------

<b>Description</b>	When set to <code>TRUE</code> , enables the experimental "job deletion nanny" feature. All job cancels will create a repeating task that will resend <code>KILL</code> signals if the initial job cancel failed. Further job cancels will be rejected with the message "job cancel in progress." This is useful for temporary failures with a job's execution node during a job delete request.
--------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

**job\_stat\_rate**

<b>Format</b>	<INTEGER>
<b>Default</b>	300 (30 in TORQUE 1.2.0p5 and earlier)
<b>Description</b>	If the mother superior has not sent an update by the specified time, at the specified time pbs_server requests an update on job status from the mother superior.

**job\_start\_timeout**

<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Specifies the pbs_server to pbs_mom TCP socket timeout in seconds that is used when the pbs_server sends a job start to the pbs_mom. It is useful when the MOM has extra overhead involved in starting jobs. If not specified, then the <a href="#">tcp_timeout</a> parameter is used.

**job\_sync\_timeout**

<b>Format</b>	<INTEGER>
<b>Default</b>	60
<b>Description</b>	When a stray job is reported on multiple nodes, the server sends a kill signal to one node at a time. This timeout determines how long the server waits between kills if the job is still being reported on any nodes.

**keep\_completed**

<b>Format</b>	<INTEGER>
<b>Default</b>	---
	 If you ran <code>torque.setup</code> on TORQUE installation, the default is 300.
<b>Description</b>	The amount of time a job will be kept in the queue after it has entered the completed state. <code>keep_completed</code> <i>must</i> be set for job dependencies to work. For more information, see <a href="#">Keeping completed jobs on page 2600</a> .

lock_file	
<b>Format</b>	<STRING>
<b>Default</b>	torque/server_priv/server.lock
<b>Description</b>	<p>Specifies the name and location of the lock file used to determine which high availability server should be active.</p> <p>If a full path is specified, it is used verbatim by TORQUE. If a relative path is specified, TORQUE will prefix it with <code>torque/server_priv</code>.</p>

lock_file_update_time	
<b>Format</b>	<INTEGER>
<b>Default</b>	3
<b>Description</b>	Specifies how often (in seconds) the thread will update the lock file.

lock_file_check_time	
<b>Format</b>	<INTEGER>
<b>Default</b>	9
<b>Description</b>	Specifies how often (in seconds) a high availability server will check to see if it should become active.

log_events	
<b>Format</b>	Bitmap
<b>Default</b>	---

## log\_events

**Description**

By default, all events are logged. However, you can customize things so that only certain events show up in the log file. These are the bitmaps for the different kinds of logs:

```
#define PBSEVENT_ERROR 0x0001 /* internal errors */
#define PBSEVENT_SYSTEM 0x0002 /* system (server) events */
#define PBSEVENT_ADMIN 0x0004 /* admin events */
#define PBSEVENT_JOB 0x0008 /* job related events */
#define PBSEVENT_JOB_USAGE 0x0010 /* End of Job accounting */
#define PBSEVENT_SECURITY 0x0020 /* security violation events */
#define PBSEVENT_SCHED 0x0040 /* scheduler events */
#define PBSEVENT_DEBUG 0x0080 /* common debug messages */
#define PBSEVENT_DEBUG2 0x0100 /* less needed debug messages */
#define PBSEVENT_FORCE 0x8000 /* set to force a message */
```

If you want to log only error, system, and job information, use `qmgr` to set `log_events` to 11:

```
set server log_events = 11
```

## log\_file\_max\_size

**Format**

&lt;INTEGER&gt;

**Default**

0

**Description**

Specifies a soft limit, in kilobytes, for the server's log file. The file size is checked every 5 minutes, and if the *current day* file size is greater than or equal to this value then it will be rolled from *X* to *X.1* and a new empty log will be opened. Any value less than or equal to 0 will be ignored by `pbs_server` (the log will not be rolled).

## log\_file\_roll\_depth

**Format**

&lt;INTEGER&gt;

**Default**

1

**Description**

Controls how deep the current day log files will be rolled, if `log_file_max_size` is set, before they are deleted.

## log\_keep\_days

**Format**

&lt;INTEGER&gt;

log_keep_days	
<b>Default</b>	0
<b>Description</b>	Specifies how long (in days) a server or MOM log should be kept.

log_level	
<b>Format</b>	<INTEGER>
<b>Default</b>	0
<b>Description</b>	Specifies the <code>pbs_server</code> logging verbosity. Maximum value is 7.

mail_body_fmt	
<b>Format</b>	A printf-like format string
<b>Default</b>	PBS Job Id: %i Job Name: %j Exec host: %h %m %d
<b>Description</b>	<p>Override the default format for the body of outgoing mail messages. A number of printf-like format specifiers and escape sequences can be used:</p> <ul style="list-style-type: none"> <li>\n new line</li> <li>\t tab</li> <li>\\ backslash</li> <li>\' single quote</li> <li>\" double quote</li> <li>%d details concerning the message</li> <li>%h PBS host name</li> <li>%i PBS job identifier</li> <li>%j PBS job name</li> <li>%m long reason for message</li> <li>%r short reason for message</li> <li>%% a single %</li> </ul>

mail_domain	
<b>Format</b>	<STRING>
<b>Default</b>	---

## mail\_domain

<b>Description</b>	Override the default domain for outgoing mail messages. If set, emails will be addressed to <user>@<hostdomain>. If unset, the job's Job_Owner attribute will be used. If set to <code>never</code> , TORQUE will never send emails.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## mail\_from

<b>Format</b>	<STRING>
---------------	----------

<b>Default</b>	adm
----------------	-----

<b>Description</b>	Specify the name of the sender when TORQUE sends emails.
--------------------	----------------------------------------------------------

## mail\_subject\_fmt

<b>Format</b>	A printf-like format string
---------------	-----------------------------

<b>Default</b>	PBS JOB %i
----------------	------------

<b>Description</b>	Override the default format for the subject of outgoing mail messages. A number of printf-like format specifiers and escape sequences can be used: \n new line \t tab \\ backslash \' single quote \" double quote %d details concerning the message %h PBS host name %i PBS job identifier %j PBS job name %m long reason for message %r short reason for message %% a single %
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## managers

<b>Format</b>	<user>@<host.sub.domain>[,<user>@<host.sub.domain>...]
---------------	--------------------------------------------------------

<b>Default</b>	root@localhost
----------------	----------------

## managers

<b>Description</b>	List of users granted batch administrator privileges. The host, sub-domain, or domain name may be wildcarded by the use of an asterisk character (*). Requires full manager privilege to set or alter.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## max\_job\_array\_size

<b>Format</b>	<INTEGER>
<b>Default</b>	Unlimited
<b>Description</b>	Sets the maximum number of jobs that can be in a single job array.

## max\_slot\_limit

<b>Format</b>	<INTEGER>
<b>Default</b>	Unlimited
<b>Description</b>	<p>This is the maximum number of jobs that can run concurrently in any job array. Slot limits can be applied at submission time with <a href="#">qsub</a>, or it can be modified with <a href="#">qalter</a>.</p> <pre>qmgr -c 'set server max_slot_limit=10'</pre> <p>No array can request a slot limit greater than 10. Any array that does not request a slot limit receives a slot limit of 10. Using the example above, slot requests greater than 10 are rejected with the message: "Requested slot limit is too large, limit is 10."</p>

## max\_threads

<b>Format</b>	<INTEGER>
<b>Default</b>	The value of min_threads $((2 * \text{the number of procs listed in } /proc/cpuinfo) + 1) * 20$
<b>Description</b>	This is the maximum number of threads that should exist in the thread pool at any time. See <a href="#">Setting min_threads and max_threads on page 2652</a> for more information.

## max\_user\_queuable

<b>Format</b>	<INTEGER>
---------------	-----------

max_user_queuable	
<b>Default</b>	Unlimited
<b>Description</b>	<p>When set, <code>max_user_queuable</code> places a system-wide limit on the amount of jobs that an individual user can queue.</p> <pre>qmgr -c 'set server max_user_queuable=500'</pre>

min_threads	
<b>Format</b>	<INTEGER>
<b>Default</b>	(2 * the number of procs listed in <code>/proc/cpuinfo</code> ) + 1. If TORQUE is unable to read <code>/proc/cpuinfo</code> , the default is 10.
<b>Description</b>	This is the minimum number of threads that should exist in the thread pool at any time. See <a href="#">Setting min_threads and max_threads on page 2652</a> for more information.

moab_array_compatible	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	This parameter places a hold on jobs that exceed the <a href="#">slot limit</a> in a job array. When one of the active jobs is completed or deleted, one of the held jobs goes to a queued state.

mom_job_sync	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	<p>When set to <code>TRUE</code>, specifies that the <code>pbs_server</code> will synchronize its view of the job queue and resource allocation with compute nodes as they come online. If a job exists on a compute node, it will be automatically cleaned up and purged. (Enabled by default in TORQUE 2.2.0 and higher.)</p> <p>Jobs that are no longer reported by the mother superior are automatically purged by <code>pbs_server</code>. Jobs that <code>pbs_server</code> instructs the MOM to cancel have their processes killed in addition to being deleted (instead of leaving them running as in versions of TORQUE prior to 4.1.1).</p>

next_job_number	
<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	<p>Specifies the ID number of the next job. If you set your job number too low and TORQUE repeats a job number that it has already used, the job will fail. Before setting <code>next_job_number</code> to a number lower than any number that TORQUE has already used, you must clear out your <code>.e</code> and <code>.o</code> files.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> If you use Moab Workload Manager and have configured it to synchronize job IDs with TORQUE), then Moab will generate the job ID and <code>next_job_number</code> will have no effect on the job ID. See <a href="#">Resource Manager Configuration on page 588</a> for more information.</p> </div>

node_check_rate	
<b>Format</b>	<INTEGER>
<b>Default</b>	600
<b>Description</b>	Specifies the minimum duration (in seconds) that a node can fail to send a status update before being marked down by the <code>pbs_server</code> daemon.

node_pack	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	Controls how multiple processor nodes are allocated to jobs. If this attribute is set to <code>TRUE</code> , jobs will be assigned to the multiple processor nodes with the fewest free processors. This packs jobs into the fewest possible nodes leaving multiple processor nodes free for jobs which need many processors on a node. If set to <code>false</code> , jobs will be scattered across nodes reducing conflicts over memory between jobs. If unset, the jobs are packed on nodes in the order that the nodes are declared to the server (in the nodes file). Default value: unset - assigned to nodes as nodes in order that were declared.

node_ping_rate	
<b>Format</b>	<INTEGER>

## node\_ping\_rate

<b>Default</b>	300
<b>Description</b>	Specifies the maximum interval (in seconds) between successive "pings" sent from the <code>pbs_server</code> daemon to the <code>pbs_mom</code> daemon to determine node/daemon health.

## no\_mail\_force

<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <code>TRUE</code> , eliminates all e-mails when <code>mail_options</code> (see <a href="#">qsub on page 2744</a> ) is set to "n". The job owner won't receive e-mails when a job is deleted by a different user or a job failure occurs. If <code>no_mail_force</code> is unset or is <code>FALSE</code> , then the job owner receives e-mails when a job is deleted by a different user or a job failure occurs.

## np\_default

<b>Format</b>	<INTEGER>
<b>Default</b>	---
<b>Description</b>	Allows the administrator to unify the number of processors ( <code>np</code> ) on all nodes. The value can be dynamically changed. A value of 0 tells <code>pbs_server</code> to use the value of <code>np</code> found in the nodes file. The maximum value is 32767.

## operators

<b>Format</b>	<user>@<host.sub.domain>[,<user>@<host.sub.domain>...]
<b>Default</b>	root@localhost
<b>Description</b>	List of users granted batch operator privileges. Requires full manager privilege to set or alter.

## pass\_cpuclock

<b>Format</b>	<BOOLEAN>
---------------	-----------

pass_cpuclock	
<b>Default</b>	<b>TRUE</b>
<b>Description</b>	<p>If set to <b>TRUE</b>, the <code>pbs_server</code> daemon passes the option and its value to the <code>pbs_mom</code> daemons for direct implementation by the daemons, making the CPU frequency adjustable as part of a <a href="#">resource request</a> by a job submission.</p> <p>If set to <b>FALSE</b>, the <code>pbs_server</code> daemon creates and passes a <code>PBS_CPUCLOCK</code> job environment variable to the <code>pbs_mom</code> daemons that contains the value of the <code>cpuclock</code> attribute used as part of a resource request by a job submission. The CPU frequencies on the MOMs are not adjusted. The environment variable is for use by prologue and epilogue scripts, enabling administrators to log and research when users are making <code>cpuclock</code> requests, as well as researchers and developers to perform CPU clock frequency changes using a method outside of that employed by the TORQUE <code>pbs_mom</code> daemons.</p>

poll_jobs	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>TRUE</b> (FALSE in TORQUE 1.2.0p5 and earlier)
<b>Description</b>	<p>If set to <b>TRUE</b>, <code>pbs_server</code> will poll job info from MOMs over time and will not block on handling requests which require this job information.</p> <p>If set to <b>FALSE</b>, no polling will occur and if requested job information is stale, <code>pbs_server</code> may block while it attempts to update this information. For large systems, this value should be set to <b>TRUE</b>.</p>

query_other_jobs	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	When set to <b>TRUE</b> , specifies whether or not non-admin users may view jobs they do not own.

record_job_info	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	This must be set to <b>TRUE</b> in order for job logging to be enabled.

record_job_script	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	<b>FALSE</b>
<b>Description</b>	<p>If set to <code>TRUE</code>, this adds the contents of the script executed by a job to the log.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;"> <p> For <code>record_job_script</code> to take effect, <a href="#">record_job_info</a> on page 2781 must be set to <code>TRUE</code>.</p> </div>

resources_available	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	<p>Allows overriding of detected resource quantity limits (see <a href="#">Assigning queue resource limits on page 2636</a>). <code>pbs_server</code> must be restarted for changes to take effect. Also, <code>resources_available</code> is constrained by the smallest of <code>queue.resources_available</code> and the <code>server.resources_available</code>.</p>

scheduling	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	---
<b>Description</b>	<p>Allows <code>pbs_server</code> to be scheduled. When <code>FALSE</code>, <code>pbs_server</code> is a resource manager that works on its own. When <code>TRUE</code>, TORQUE allows a scheduler, such as Moab or Maui, to dictate what <code>pbs_server</code> should do.</p>

submit_hosts	
<b>Format</b>	"<HOSTNAME>[,<HOSTNAME>]..."
<b>Default</b>	---
<b>Description</b>	<p>Indicates which hosts included in the server nodes file located at <code>\$TORQUE/server_priv/nodes</code> (see <a href="#">Server node file configuration on page 2572</a>) can submit batch or interactive jobs (see <a href="#">Configuring job submission hosts on page 2566</a>). For more information on adding hosts that are not included in the first nodes file, see the <a href="#">acl_hosts</a> parameter.</p>

tcp_timeout	
<b>Format</b>	<INTEGER>
<b>Default</b>	300
<b>Description</b>	<p>Specifies the timeout for idle TCP connections. If no communication is received by the server on the connection after the timeout, the server closes the connection. There is an exception for connections made to the server on port 15001 (default); timeout events are ignored on the server for such connections established by a client utility or scheduler. Responsibility rests with the client to close the connection first (See <a href="#">Appendix F: Large cluster considerations on page 2811</a> for additional information.).</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> If you use Moab Workload Manager, prevent communication errors by giving <b>tcp_timeout</b> at least twice the value of the Moab <a href="#">RMPOLLINTERVAL</a>.</p> </div>

thread_idle_seconds	
<b>Format</b>	<INTEGER>
<b>Default</b>	300
<b>Description</b>	<p>This is the number of seconds a thread can be idle in the thread pool before it is deleted. If threads should not be deleted, set to -1. TORQUE will always maintain at least <a href="#">min_threads</a> number of threads, even if all are idle.</p>

## Appendix C: Node manager (MOM) configuration

Under TORQUE, MOM configuration is accomplished using the `mom_priv/config` file located in the PBS directory on each execution server. You must create this file and insert any desired lines in a text editor (blank lines are allowed). When you modify the `mom_priv/config` file, you must restart `pbs_mom`.

The following examples demonstrate two methods of modifying the `mom_priv/config` file:

```
> echo "\$loglevel 3" > /var/spool/torque/mom_priv/config
```

```
> vim /var/spool/torque/mom_priv/config
...
$loglevel 3
```

For details, see these topics:

- [Parameters on page 2784](#)
- [Node features and generic consumable resource specification on page 2801](#)

- [Command-line arguments on page 2801](#)

#### Related topics

- [Appendix A: Commands overview on page 2686](#)
- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Parameters

These parameters go in the `mom_priv/config` file. They control various behaviors for the MOMs.

arch	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the architecture of the local machine. This information is used by the scheduler only.
<b>Example</b>	<code>arch ia64</code>

\$attempt_to_make_dir	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	<p>When set to <i>TRUE</i>, specifies that you want TORQUE to attempt to create the output directories for jobs if they do not already exist. Default is <i>FALSE</i>.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> TORQUE uses this parameter to make the directory as the <i>user</i> and not as <i>root</i>. TORQUE will create the directory (or directories) <b>ONLY</b> if the user has permissions to do so.</p> </div>
<b>Example</b>	<code>\$attempt_to_make_dir true</code>

\$clienthost	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies the machine running <code>pbs_server</code>.</p> <div style="border: 1px solid red; padding: 5px; margin-top: 10px;"> <p> This parameter is deprecated. Use <a href="#">\$pbsserver</a>.</p> </div>
<b>Example</b>	<code>\$clienthost node01.teracluster.org</code>

\$check_poll_time	
<b>Format</b>	<STRING>
<b>Description</b>	Amount of time between checking running jobs, polling jobs, and trying to resend obituaries for jobs that haven't sent successfully. Default is 45 seconds.
<b>Example</b>	<code>\$check_poll_time 90</code>

\$configversion	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the version of the config file data.
<b>Example</b>	<code>\$configversion 113</code>

\$cputmult	
<b>Format</b>	<FLOAT>
<b>Description</b>	CPU time multiplier. <div style="border: 1px solid #0070c0; border-radius: 10px; padding: 5px; margin-top: 10px;">  If set to 0.0, MOM level cputime enforcement is disabled.         </div>
<b>Example</b>	<code>\$cputmult 2.2</code>

\$down_on_error	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Causes the MOM to report itself as state "down" to <code>pbs_server</code> in the event of a failed health check. This feature is experimental. For more information, see <a href="#">Parameters on page 2784</a> .
<b>Example</b>	<code>\$down_on_error true</code>

**\$enablemomrestart**

<b>Format</b>	<BOOLEAN>
<b>Description</b>	Enables automatic restarts of the MOM. If enabled, the MOM will check if its binary has been updated and restart itself at a safe point when no jobs are running; thus making upgrades easier. The check is made by comparing the mtime of the <code>pbs_mom</code> executable. Command-line args, the process name, and the PATH env variable are preserved across restarts. It is recommended that this not be enabled in the config file, but enabled when desired with <code>momctl</code> (see <a href="#">Parameters on page 2784</a> for more information.)
<b>Example</b>	<code>\$enablemomrestart true</code>

**\$exec\_with\_exec**

<b>Format</b>	<BOOLEAN>
<b>Description</b>	<code>pbs_mom</code> uses the <code>exec</code> command to start the job script rather than the TORQUE default method, which is to pass the script's contents as the input to the shell. This means that if you trap signals in the job script, they will be trapped for the job. Using the default method, you would need to configure the shell to also trap the signals. Default is FALSE.
<b>Example</b>	<code>\$exec_with_exec true</code>

**\$ext\_pwd\_retry**

<b>Format</b>	<INTEGER>
<b>Description</b>	(Available in TORQUE 2.5.10, 3.0.4, and later.) Specifies the number of times to retry checking the password. Useful in cases where external password validation is used, such as with LDAP. The default value is 3 retries.
<b>Example</b>	<code>\$ext_pwd_retry = 5</code>

**\$ideal\_load**

<b>Format</b>	<FLOAT>
<b>Description</b>	Ideal processor load.
<b>Example</b>	<code>\$ideal_load 4.0</code>

\$igncput	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Ignores limit violation pertaining to CPU time. Default is FALSE.
<b>Example</b>	\$igncput true

\$ignmem	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Ignores limit violations pertaining to physical memory. Default is FALSE.
<b>Example</b>	\$ignmem true

\$ignvmem	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Ignores limit violations pertaining to virtual memory. Default is FALSE.
<b>Example</b>	\$ignvmem true

\$ignwalltime	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	Ignore walltime (do not enable MOM based walltime limit enforcement).
<b>Example</b>	\$ignwalltime true

\$jobdirectory_sticky	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	When this option is set ( <code>true</code> ), the job directory on the MOM can have a sticky bit set. The default is <code>false</code> .

**\$jobdirectory\_sticky**

<b>Example</b>	<code>\$jobdirectory_sticky true</code>
----------------	-----------------------------------------

**\$job\_exit\_wait\_time**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Description</b>	This is the timeout to clean up parallel jobs after one of the sister nodes for the parallel job goes down or is otherwise unresponsive. The MOM sends out all of its kill job requests to sisters and marks the time. Additionally, the job is placed in the substate <code>JOB_SUBSTATE_EXIT_WAIT</code> . The MOM then periodically checks jobs in this state and if they are in this state for more than the specified time, death is assumed and the job gets cleaned up. Default is 10 minutes.
--------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$job_exit_wait_time 300</code>
----------------	---------------------------------------

**\$job\_output\_file\_unmask**

<b>Format</b>	<STRING>
---------------	----------

<b>Description</b>	Uses the specified umask when creating job output and error files. Values can be specified in base 8, 10, or 16; leading 0 implies octal and leading 0x or 0X hexadecimal. A value of "userdefault" will use the user's default umask. This parameter is in version 2.3.0 and later.
--------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$job_output_file_umask 027</code>
----------------	------------------------------------------

**\$job\_starter**

<b>Format</b>	<STRING>
---------------	----------

<b>Description</b>	Specifies the fully qualified pathname of the job starter. If this parameter is specified, instead of executing the job command and job arguments directly, the MOM will execute the job starter, passing the job command and job arguments to it as its arguments. The job starter can be used to launch jobs within a desired environment.
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<pre> \$job_starter /var/torque/mom_priv/job_starter.sh &gt; cat /var/torque/mom_priv/job_starter.sh #!/bin/bash export FOOHOME=/home/foo ulimit -n 314 \$*</pre>
----------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------

\$log_directory	
<b>Format</b>	<STRING>
<b>Description</b>	Changes the log directory. Default is TORQUE_HOME/mom_logs/. TORQUE_HOME default is /var/spool/torque/ but can be changed in the ./configure script. The value is a string and should be the full path to the desired MOM log directory.
<b>Example</b>	<code>\$log_directory /opt/torque/mom_logs/</code>

\$log_file_suffix	
<b>Format</b>	<STRING>
<b>Description</b>	Optional suffix to append to log file names. If %h is the suffix, pbs_mom appends the hostname for where the log files are stored if it knows it, otherwise it will append the hostname where the MOM is running.
<b>Example</b>	<code>\$log_file_suffix %h = 20100223.mybox</code> <code>\$log_file_suffix foo = 20100223.foo</code>

\$logevent	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies a bitmap for event types to log.
<b>Example</b>	<code>\$logevent 255</code>

\$loglevel	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies the verbosity of logging with higher numbers specifying more verbose logging. Values may range between 0 and 7.
<b>Example</b>	<code>\$loglevel 4</code>

**\$log\_file\_max\_size**

<b>Format</b>	<INTEGER>
<b>Description</b>	Soft limit for log file size in kilobytes. Checked every 5 minutes. If the log file is found to be greater than or equal to <code>log_file_max_size</code> the current log file will be moved from X to X.1 and a new empty file will be opened.
<b>Example</b>	<code>\$log_file_max_size = 100</code>

**\$log\_file\_roll\_depth**

<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies how many times a log file will be rolled before it is deleted.
<b>Example</b>	<code>\$log_file_roll_depth = 7</code>

**\$log\_keep\_days**

<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies how many days to keep log files. <code>pbs_mom</code> deletes log files older than the specified number of days. If not specified, <code>pbs_mom</code> won't delete log files based on their age.
<b>Example</b>	<code>\$log_keep_days 10</code>

**\$max\_conn\_timeout\_micro\_sec**

<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies how long <code>pbs_mom</code> should wait for a connection to be made. Default value is 10000 (.1 sec).
<b>Example</b>	<code>\$max_conn_timeout_micro_sec 30000</code> This sets the connection timeout on the MOM to .3 seconds.

**\$max\_join\_job\_wait\_time**

<b>Format</b>	<INTEGER>
<b>Description</b>	The interval to wait for jobs stuck in a prerun state before deleting them from the MOMs and requeueing them on the server. Default is 10 minutes.
<b>Example</b>	\$max_join_job_wait_time 300

**\$max\_load**

<b>Format</b>	<FLOAT>
<b>Description</b>	Maximum processor load.
<b>Example</b>	\$max_load 4.0

**\$memory\_pressure\_duration**

<b>Format</b>	<INTEGER>
<b>Description</b>	<i>(Applicable in version 3.0 and later.)</i> Memory pressure duration sets a limit to the number of times the value of memory_pressure_threshold can be exceeded before a process is terminated. This can only be used with <a href="#">\$memory_pressure_threshold</a> .
<b>Example</b>	\$memory_pressure_duration 5

**\$memory\_pressure\_threshold**

<b>Format</b>	<INTEGER>
---------------	-----------

**\$memory\_pressure\_threshold**

<b>Description</b>	<p>(Applicable in version 3.0 and later.) The memory_pressure of a cpuset provides a simple per-cpuset running average of the rate that the processes in a cpuset are attempting to free up in-use memory on the nodes of the cpuset to satisfy additional memory requests. The memory_pressure_threshold is an integer number used to compare against the reclaim rate provided by the memory_pressure file. If the threshold is exceeded and memory_pressure_duration is set, then the process terminates after exceeding the threshold by the number of times set in memory_pressure_duration. If memory_pressure duration is not set, then a warning is logged and the process continues. Memory_pressure_threshold is only valid with memory_pressure enabled in the root cpuset.</p> <p>To enable, log in as the super user and execute the command <code>echo 1 &gt;&gt; /dev/cpuset/memory_pressure_enabled</code>. See the cpuset man page for more information concerning memory pressure.</p>
<b>Example</b>	<code>\$memory_pressure_threshold 1000</code>

**\$mom\_hierarchy\_retry\_time**

<b>Format</b>	<SECONDS>
<b>Description</b>	Specifies the amount of time that a MOM waits to retry a node in the hierarchy path after a failed connection to that node. The default is 90 seconds.
<b>Example</b>	<code>\$mom_hierarchy_retry_time 30</code>

**\$mom\_host**

<b>Format</b>	<STRING>
<b>Description</b>	Sets the local hostname as used by pbs_mom.
<b>Example</b>	<code>\$mom_host node42</code>

**\$node\_check\_script**

<b>Format</b>	<STRING>
<b>Description</b>	Specifies the fully qualified pathname of the health check script to run (see <a href="#">Compute node health check on page 2677</a> for more information).
<b>Example</b>	<code>\$node_check_script /opt/batch_tools/nodecheck.pl</code>

\$node_check_interval	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies the number of MOM intervals between subsequent executions of the specified health check. This value default to 1 indicating the check is run every MOM interval (see <a href="#">Compute node health check on page 2677</a> for more information).</p> <p>\$node_check_interval has two special strings that can be set:</p> <ul style="list-style-type: none"> <li>• <i>jobstart</i> – makes the node health script run when a job is started.</li> <li>• <i>jobend</i> – makes the node health script run after each job has completed on a node.</li> </ul>
<b>Example</b>	<code>\$node_check_interval 5</code>

\$nodefile_suffix	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the suffix to append to a host names to denote the data channel network adapter in a multi-homed compute node.
<b>Example</b>	<p><code>\$nodefile_suffix i</code></p> <p>with the suffix of "i" and the control channel adapter with the name <i>node01</i>, the data channel would have a hostname of <i>node01i</i>.</p>

\$nospool_dir_list	
<b>Format</b>	<STRING>
<b>Description</b>	<p>If this is configured, the job's output is spooled in the working directory of the job or the specified output directory.</p> <p>Specify the list in full paths, delimited by commas. If the job's working directory (or specified output directory) is in one of the paths in the list (or a subdirectory of one of the paths in the list), the job is spooled directly to the output location. \$nospool_dir_list * is accepted.</p> <p>The user that submits the job must have write permission on the folder where the job is written, and read permission on the folder where the file is spooled.</p> <p>Alternatively, you can use the \$spool_as_final_name parameter to force the job to spool directly to the final output.</p> <div style="border: 1px solid #0070c0; border-radius: 10px; padding: 10px; margin-top: 10px;"> <p> This should generally be used only when the job can run on the same machine as where the output file goes, or if there is a shared filesystem. If not, this parameter can slow down the system or fail to create the output file.</p> </div>
<b>Example</b>	<code>\$nospool_dir_list /home/mike/jobs/,/var/tmp/spool/</code>

opsys	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the operating system of the local machine. This information is used by the scheduler only.
<b>Example</b>	<code>opsys RHEL3</code>

\$pbsclient	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies machines which the MOM daemon will trust to run resource manager commands via <a href="#">momctl</a> . This may include machines where monitors, schedulers, or admins require the use of this command.
<b>Example</b>	<code>\$pbsclient node01.teracluster.org</code>

\$pbserver	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the machine running pbs_server. <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;">  This parameter replaces the deprecated parameter <a href="#">\$clienthost</a>. </div>
<b>Example</b>	<code>\$pbserver node01.teracluster.org</code>

\$prologalarm	
<b>Format</b>	<INTEGER>
<b>Description</b>	Specifies maximum duration (in seconds) which the MOM will wait for the job prologue or job epilogue to complete. The default value is 300 seconds (5 minutes). The maximum value is 300 and when set to anything higher than that, it is treated as 300.
<b>Example</b>	<code>\$prologalarm 60</code>

\$rcpcmd	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the full path and optional additional command line args to use to perform remote copies.
<b>Example</b>	<pre>mom_priv/config: \$rcpcmd /usr/local/bin/scp -i /etc/sshauth.dat</pre>

\$remote_reconfig	
<b>Format</b>	<STRING>
<b>Description</b>	Enables the ability to remotely reconfigure pbs_mom with a new config file. Default is disabled. This parameter accepts various forms of true, yes, and 1. For more information on how to reconfigure MOMs, see <a href="#">momctl-r</a> .
<b>Example</b>	<pre>\$remote_reconfig true</pre>

\$remote_checkpoint_dirs	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies which server checkpoint directories are remotely mounted. It tells the MOM which directories are shared with the server. Using remote checkpoint directories eliminates the need to copy the checkpoint files back and forth between the MOM and the server. All entries must be on the same line, separated by a space.
<b>Example</b>	<pre>\$remote_checkpoint_dirs /checkpointFiles /bigStorage /fast</pre> <p><i>This informs the MOM that the /checkpointFiles, /bigStorage, and /fast directories are remotely mounted checkpoint directories.</i></p>

\$reduce_prolog_checks	
<b>Format</b>	<STRING>
<b>Description</b>	If enabled, TORQUE will only check if the file is a regular file and is executable, instead of the normal checks listed on the prologue and epilogue page. Default is FALSE.

**\$reduce\_prolog\_checks**

<b>Example</b>	<code>\$reduce_prolog_checks true</code>
----------------	------------------------------------------

**\$reject\_job\_submission**

<b>Format</b>	<BOOLEAN>
---------------	-----------

<b>Description</b>	If set to <b>TRUE</b> , jobs will be rejected and the user will receive the message, "Jobs cannot be run on mom %s." Default is FALSE.
--------------------	----------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$reject_job_submission job01</code>
----------------	--------------------------------------------

**\$resend\_join\_job\_wait\_time**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Description</b>	This is the timeout for the Mother Superior to re-send the join job request if it didn't get a reply from all the sister MOMs. The resend happens only once. Default is 5 minutes.
--------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$resend_join_job_wait_time 120</code>
----------------	----------------------------------------------

**\$restricted**

<b>Format</b>	<STRING>
---------------	----------

<b>Description</b>	Specifies hosts which can be trusted to access MOM services as non-root. By default, no hosts are trusted to access MOM services as non-root.
--------------------	-----------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$restricted *.teracluster.org</code>
----------------	---------------------------------------------

**\$rpp\_throttle**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Description</b>	This integer is in microseconds and causes a sleep after every RPP packet is sent. It is for systems that experience job failures because of incomplete data.
--------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$rpp_throttle 100</code> (will cause a 100 microsecond sleep)
----------------	-------------------------------------------------------------------------

size[fs=<FS>]	
<b>Format</b>	N/A
<b>Description</b>	<p>Specifies that the available and configured disk space in the &lt;FS&gt; filesystem is to be reported to the pbs_server and scheduler.</p> <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;"> <p><b>i</b> To request disk space on a per job basis, specify the file resource as in <code>qsub -l nodes=1, file=1000kb</code>.</p> </div> <div style="border: 1px solid #ccc; padding: 5px; margin: 5px 0;"> <p><b>i</b> Unlike most MOM config options, the <code>size</code> parameter is not preceded by a "\$" character.</p> </div>
<b>Example</b>	<pre>size[fs=/localscratch]</pre> <p>The available and configured disk space in the <code>/localscratch</code> filesystem will be reported.</p>

\$source_login_batch	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies whether or not MOM will source the <code>/etc/profile</code>, etc. type files for <i>batch</i> jobs. Parameter accepts various forms of true, false, yes, no, 1 and 0. Default is TRUE. This parameter is in version 2.3.1 and later.</p>
<b>Example</b>	<pre>\$source_login_batch False</pre> <p>MOM will bypass the sourcing of <code>/etc/profile</code>, etc. type files.</p>

\$source_login_interactive	
<b>Format</b>	<STRING>
<b>Description</b>	<p>Specifies whether or not MOM will source the <code>/etc/profile</code>, etc. type files for <i>interactive</i> jobs. Parameter accepts various forms of true, false, yes, no, 1 and 0. Default is TRUE. This parameter is in version 2.3.1 and later.</p>
<b>Example</b>	<pre>\$source_login_interactive False</pre> <p>MOM will bypass the sourcing of <code>/etc/profile</code>, etc. type files.</p>

\$spool_as_final_name	
<b>Format</b>	<BOOLEAN>

**\$spool\_as\_final\_name**

<b>Description</b>	This makes the job write directly to its output destination instead of a spool directory. This allows users easier access to the file if they want to watch the jobs output as it runs.
--------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<code>\$spool_as_final_name true</code>
----------------	-----------------------------------------

**\$status\_update\_time**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Description</b>	Specifies the number of seconds between subsequent MOM-to-server update reports. Default is 45 seconds.
--------------------	---------------------------------------------------------------------------------------------------------

<b>Example</b>	<pre>status_update_time: \$status_update_time 120</pre> <p>MOM will send server update reports every 120 seconds.</p>
----------------	-----------------------------------------------------------------------------------------------------------------------

**\$thread\_unlink\_calls**

<b>Format</b>	<BOOLEAN>
---------------	-----------

<b>Description</b>	Threads calls to unlink when deleting a job. Default is false. If it is set to TRUE, pbs_mom will use a thread to delete the job's files.
--------------------	-------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<pre>thread_unlink_calls: \$thread_unlink_calls true</pre>
----------------	------------------------------------------------------------

**\$timeout**

<b>Format</b>	<INTEGER>
---------------	-----------

<b>Description</b>	<p>Specifies the number of seconds before a TCP connection on the MOM will timeout. Default is 300 seconds.</p> <p>In version 3.x and earlier, this specifies the number of seconds before MOM-to-MOM messages will timeout if RPP is disabled. Default is 60 seconds.</p>
--------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

<b>Example</b>	<pre>\$timeout 120</pre> <p>A TCP connection will wait up to 120 seconds before timing out.</p> <p>For 3.x and earlier, MOM-to-MOM communication will allow up to 120 seconds before timing out.</p>
----------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

\$tmpdir	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies a directory to create job-specific scratch space (see <a href="#">Creating Per-Job Temporary Directories</a> ).
<b>Example</b>	\$tmpdir /localscratch

\$usecp	
<b>Format</b>	<HOST>:<SRCDIR> <DSTDIR>
<b>Description</b>	Specifies which directories should be staged (see <a href="#">NFS and other networked filesystems on page 2656</a> )
<b>Example</b>	\$usecp *.fte.com:/data /usr/local/data

\$use_smt	
<b>Format</b>	<BOOLEAN>
<b>Description</b>	<p>Indicates that the user would like to use SMT. If set, each logical core inside of a physical core will be used as a normal core for cpusets. This parameter is on by default.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p> If SMT is used, you will need to set the <i>np</i> attribute so that each logical processor is counted.</p> </div>
<b>Example</b>	\$use_smt false

\$varattr	
<b>Format</b>	<INTEGER> <STRING>

\$varattr	
<b>Description</b>	<p>Provides a way to keep track of dynamic attributes on nodes.</p> <p>&lt;INTEGER&gt; is how many seconds should go by between calls to the script to update the dynamic values. If set to -1, the script is read only one time.</p> <p>&lt;STRING&gt; is the script path. This script should check for whatever dynamic attributes are desired, and then output lines in this format:</p> <pre>name=value</pre> <p>Include any arguments after the script's full path. These features are visible in the output of <a href="#">pbsnodes -a</a></p> <pre>varattr=Matlab=7.1;Octave=1.0.</pre> <p>For information about using \$varattr to request dynamic features in Moab, see <a href="#">Resource Manager Extensions on page 618</a>.</p>
<b>Example</b>	<code>\$varattr 25 /usr/local/scripts/nodeProperties.pl arg1 arg2 arg3</code>

\$wallmult	
<b>Format</b>	<FLOAT>
<b>Description</b>	<p>Sets a factor to adjust walltime usage by multiplying a default job time to a common reference system. It modifies real walltime on a per-MOM basis (MOM configuration parameters). The factor is used for walltime calculations and limits in the same way that cputmult is used for cpu time.</p> <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 5px; margin-top: 10px;"> <p> If set to 0.0, MOM level walltime enforcement is disabled.</p> </div>
<b>Example</b>	<code>\$wallmult 2.2</code>

\$xauthpath	
<b>Format</b>	<STRING>
<b>Description</b>	Specifies the path to the xauth binary to enable X11 forwarding.
<b>Example</b>	<code>\$xauthpath /opt/bin/xauth/</code>

#### Related topics

- [Appendix C: Node manager \(MOM\) configuration on page 2783](#)

## Node features and generic consumable resource specification

Node features (a.k.a. "node properties") are opaque labels which can be applied to a node. They are not consumable and cannot be associated with a value. (Use generic resources described below for these purposes). Node features are configured within the nodes file on the [pbs\\_server](#) head node. This file can be used to specify an arbitrary number of node features.

Additionally, per node consumable generic resources may be specified using the format "<ATTR> <VAL>" with no leading dollar ("\$\$") character. When specified, this information is routed to the scheduler and can be used in scheduling decisions. For example, to indicate that a given host has two tape drives and one node-locked matlab license available for batch jobs, the following could be specified:

mom\_priv/config:

```
$clienthost 241.13.153.7
tape 2
matlab 1
```

Dynamic consumable resource information can be routed in by specifying a path preceded by an exclamation point (!) as in the example below. If the resource value is configured in this manner, the specified file will be periodically executed to load the effective resource value.

mom\_priv/config:

```
$clienthost 241.13.153.7
tape !/opt/rm/gettapecount.pl
matlab !/opt/tools/getlicensecount.pl
```

### Related topics

- [Appendix C: Node manager \(MOM\) configuration on page 2783](#)

## Command-line arguments

Below is a table of `pbs_mom` command-line startup flags.

Flag	Description
<b>a</b> <integer>	Alarm time in seconds.
<b>c</b> <file>	Config file path.
<b>C</b> <dir- ectory>	Checkpoint path.
<b>d</b> <dir- ectory>	Home directory.
<b>L</b> <file>	Log file.

Flag	Description
<b>M</b> <integer>	MOM port to listen on.
<b>p</b>	Perform 'poll' based job recovery on restart (jobs persist until associated processes terminate).
<b>P</b>	On restart, deletes all jobs that were running on MOM (Available in 2.4.X and later).
<b>q</b>	On restart, requeues all jobs that were running on MOM (Available in 2.4.X and later).
<b>r</b>	On restart, kills all processes associated with jobs that were running on MOM, and then requeues the jobs.
<b>R</b> <integer>	MOM 'RM' port to listen on.
<b>S</b> <integer>	pbs_server port to connect to.
<b>v</b>	Display version information and exit.
<b>x</b>	Disable use of privileged port.
<b>?</b>	Show usage information and exit.

For more details on these command-line options, see [pbs\\_mom on page 2693](#).

#### Related topics

- [Appendix C: Node manager \(MOM\) configuration on page 2783](#)

## Appendix D: Diagnostics and error codes

TORQUE has a diagnostic script to assist you in giving TORQUE Support the files they need to support issues. It should be run by a user that has access to run all TORQUE commands and access to all TORQUE directories (this is usually root).

The script (`contrib/diag/tdiag.sh`) is available in TORQUE 2.3.8, TORQUE 2.4.3, and later. The script grabs the node file, server and MOM log files, and captures the output of `qmgr -c 'p s'`. These are put in a tar file.

The script also has the following options (this can be shown in the command line by entering `./tdiag.sh -h`):

```
USAGE: ./torque_diag [-d DATE] [-h] [-o OUTPUT_FILE] [-t TORQUE_HOME]
```

- *DATE* should be in the format YYYYmmdd. For example, "20091130" would be the date for November 30th, 2009. If no date is specified, today's date is used.
- *OUTPUT\_FILE* is the optional name of the output file. The default output file is `torque_diag<today's_date>.tar.gz`. *TORQUE\_HOME* should be the path to your TORQUE directory. If no directory is specified, `/var/spool/torque` is the default.

Table 4-6: TORQUE error codes

Error code name	Number	Description
PBSE_FLOOR	15000	No error
PBSE_UNKJOBID	15001	Unknown job ID error
PBSE_NOATTR	15002	Undefined attribute
PBSE_ATTRRO	15003	Cannot set attribute, read only or insufficient permission
PBSE_IVALREQ	15004	Invalid request
PBSE_UNKREQ	15005	Unknown request
PBSE_TOOMANY	15006	Too many submit retries
PBSE_PERM	15007	Unauthorized Request
PBSE_IFF_NOT_FOUND	15008	trqauthd unable to authenticate
PBSE_MUNGE_NOT_FOUND	15009	Munge executable not found, unable to authenticate
PBSE_BADHOST	15010	Access from host not allowed, or unknown host
PBSE_JOBEXIST	15011	Job with requested ID already exists
PBSE_SYSTEM	15012	System error
PBSE_INTERNAL	15013	PBS server internal error
PBSE_REGROUTE	15014	Dependent parent job currently in routing queue
PBSE_UNKSIG	15015	Unknown/illegal signal name

Error code name	Number	Description
PBSE_BADATVAL	15016	Illegal attribute or resource value for
PBSE_MODATTRUN	15017	Cannot modify attribute while job running
PBSE_BADSTATE	15018	Request invalid for state of job
PBSE_UNKQUE	15020	Unknown queue
PBSE_BADCRED	15021	Invalid credential
PBSE_EXPIRED	15022	Expired credential
PBSE_QUNOENB	15023	Queue is not enabled
PBSE_QACCESS	15024	Access to queue is denied
PBSE_BADUSER	15025	Bad UID for job execution
PBSE_HOPCOUNT	15026	Job routing over too many hops
PBSE_QUEEXIST	15027	Queue already exists
PBSE_ATTRTYPE	15028	Incompatible type
PBSE_QUEBUSY	15029	Cannot delete busy queue
PBSE_QUENBIG	15030	Queue name too long
PBSE_NOSUP	15031	No support for requested service
PBSE_QUENOEN	15032	Cannot enable queue, incomplete definition
PBSE_PROTOCOL	15033	Batch protocol error
PBSE_BADATLST	15034	Bad attribute list structure
PBSE_NOCONNECTS	15035	No free connections
PBSE_NOSERVER	15036	No server specified

Error code name	Number	Description
PBSE_UNKRESC	15037	Unknown resource type
PBSE_EXCQRESC	15038	Job exceeds queue resource limits
PBSE_QUENODFLT	15039	No default queue specified
PBSE_NORERUN	15040	Job is not rerunnable
PBSE_ROUTEREJ	15041	Job rejected by all possible destinations (check syntax, queue resources, ...)
PBSE_ROUTEEXPD	15042	Time in Route Queue Expired
PBSE_MOMREJECT	15043	Execution server rejected request
PBSE_BADSCRIPT	15044	(qsub) cannot access script file
PBSE_STAGEIN	15045	Stage-in of files failed
PBSE_RESCUNAV	15046	Resource temporarily unavailable
PBSE_BADGRP	15047	Bad GID for job execution
PBSE_MAXQUED	15048	Maximum number of jobs already in queue
PBSE_CKPSY	15049	Checkpoint busy, may retry
PBSE_EXLIMIT	15050	Resource limit exceeds allowable
PBSE_BADACCT	15051	Invalid Account
PBSE_ALRDYEXIT	15052	Job already in exit state
PBSE_NOCOPYFILE	15053	Job files not copied
PBSE_CLEANEOUT	15054	Unknown job id after clean init
PBSE_NOSYNCMSTR	15055	No master found for sync job set

Error code name	Number	Description
PBSE_BADDEPEND	15056	Invalid Job Dependency
PBSE_DUPLIST	15057	Duplicate entry in list
PBSE_DISPROTO	15058	Bad DIS based Request Protocol
PBSE_EXECTHERE	15059	Cannot execute at specified host because of checkpoint or stagein files
PBSE_SISREJECT	15060	Sister rejected
PBSE_SISCOMM	15061	Sister could not communicate
PBSE_SVRDOWN	15062	Request not allowed: Server shutting down
PBSE_CKPSHORT	15063	Not all tasks could checkpoint
PBSE_UNKNODE	15064	Unknown node
PBSE_UNKNODEATR	15065	Unknown node-attribute
PBSE_NONODES	15066	Server has no node list
PBSE_NODENBIG	15067	Node name is too big
PBSE_NODEEXIST	15068	Node name already exists
PBSE_BADNDATVAL	15069	Illegal value for
PBSE_MUTUALEX	15070	Mutually exclusive values for
PBSE_GMODERR	15071	Modification failed for
PBSE_NORELYMOM	15072	Server could not connect to MOM
PBSE_NOTSNODE	15073	No time-share node available
PBSE_JOBTYPE	15074	Wrong job type

Error code name	Number	Description
PBSE_BADACLHOST	15075	Bad ACL entry in host list
PBSE_MAXUSERQUED	15076	Maximum number of jobs already in queue for user
PBSE_BADDISALLOWTYPE	15077	Bad type in disallowed_types list
PBSE_NOINTERACTIVE	15078	Queue does not allow interactive jobs
PBSE_NOBATCH	15079	Queue does not allow batch jobs
PBSE_NORERUNABLE	15080	Queue does not allow rerunable jobs
PBSE_NONONRERUNABLE	15081	Queue does not allow nonrerunable jobs
PBSE_UNKARRAYID	15082	Unknown Array ID
PBSE_BAD_ARRAY_REQ	15083	Bad Job Array Request
PBSE_BAD_ARRAY_DATA	15084	Bad data reading job array from file
PBSE_TIMEOUT	15085	Time out
PBSE_JOBNOTFOUND	15086	Job not found
PBSE_NOFAULTTOLERANT	15087	Queue does not allow fault tolerant jobs
PBSE_NOFAULTINTOLERANT	15088	Queue does not allow fault intolerant jobs
PBSE_NOJOBARRAYS	15089	Queue does not allow job arrays
PBSE_RELAYED_TO_MOM	15090	Request was relayed to a MOM
PBSE_MEM_MALLOC	15091	Error allocating memory - out of memory
PBSE_MUTEX	15092	Error allocating controlling mutex (lock/unlock)
PBSE_THREADATTR	15093	Error setting thread attributes
PBSE_THREAD	15094	Error creating thread

Error code name	Number	Description
PBSE_SELECT	15095	Error in socket select
PBSE_SOCKET_FAULT	15096	Unable to get connection to socket
PBSE_SOCKET_WRITE	15097	Error writing data to socket
PBSE_SOCKET_READ	15098	Error reading data from socket
PBSE_SOCKET_CLOSE	15099	Socket close detected
PBSE_SOCKET_LISTEN	15100	Error listening on socket
PBSE_AUTH_INVALID	15101	Invalid auth type in request
PBSE_NOT_IMPLEMENTED	15102	This functionality is not yet implemented
PBSE_QUEENOTAVAILABLE	15103	Queue is currently not available
PBSE_TMPDIFFOWNER	15104	tmpdir owned by another user
PBSE_TMPNOTDIR	15105	tmpdir exists but is not a directory
PBSE_TMPNONAME	15106	tmpdir cannot be named for job
PBSE_CANTOPENSOCKET	15107	Cannot open demux sockets
PBSE_CANTCONTACTSISTERS	15108	Cannot send join job to all sisters
PBSE_CANTCREATETMPDIR	15109	Cannot create tmpdir for job
PBSE_BADMOMSTATE	15110	Mom is down, cannot run job
PBSE_SOCKET_INFORMATION	15111	Socket information is not accessible
PBSE_SOCKET_DATA	15112	Data on socket does not process correctly
PBSE_CLIENT_INVALID	15113	Client is not allowed/trusted
PBSE_PREMATURE_EOF	15114	Premature End of File

Error code name	Number	Description
PBSE_CAN_NOT_SAVE_FILE	15115	Error saving file
PBSE_CAN_NOT_OPEN_FILE	15116	Error opening file
PBSE_CAN_NOT_WRITE_FILE	15117	Error writing file
PBSE_JOB_FILE_CORRUPT	15118	Job file corrupt
PBSE_JOB_RERUN	15119	Job can not be rerun
PBSE_CONNECT	15120	Can not establish connection
PBSE_JOBWORKDELAY	15121	Job function must be temporarily delayed
PBSE_BAD_PARAMETER	15122	Parameter of function was invalid
PBSE_CONTINUE	15123	Continue processing on job. (Not an error)
PBSE_JOBSTATE	15124	Current sub state does not allow trasaction.
PBSE_CAN_NOT_MOVE_FILE	15125	Error moving file
PBSE_JOB_RECYCLED	15126	Job is being recycled
PBSE_JOB_ALREADY_IN_QUEUE	15127	Job is already in destination queue.
PBSE_INVALID_MUTEX	15128	Mutex is NULL or otherwise invalid
PBSE_MUTEX_ALREADY_LOCKED	15129	The mutex is already locked by this object
PBSE_MUTEX_ALREADY_UNLOCKED	15130	The mutex has already been unlocked by this object
PBSE_INVALID_SYNTAX	15131	Command syntax invalid
PBSE_NODE_DOWN	15132	A node is down. Check the MOM and host

Error code name	Number	Description
PBSE_SERVER_NOT_FOUND	15133	Could not connect to batch server
PBSE_SERVER_BUSY	15134	Server busy. Currently no available threads

## Appendix E: Considerations before upgrading

TORQUE is flexible in regards to how it can be upgraded. In most cases, a TORQUE "shutdown" followed by a *configure, make, make install* procedure as documented in this guide is all that is required (see [Installing on page 1](#)). This process will preserve existing configuration and in most cases, existing workload.

A few considerations are included below:

- If upgrading from OpenPBS, PBSPro, or TORQUE 1.0.3 or earlier, queued jobs whether active or idle will be lost. In such situations, job queues should be completely drained of all jobs.
- If not using the `pbs_mom -r` or `-p` flag (see [Command-line arguments on page 2801](#)), running jobs may be lost. In such cases, running jobs should be allowed to be completed or should be requeued before upgrading TORQUE.
- `pbs_mom` and `pbs_server` daemons of differing versions may be run together. However, not all combinations have been tested and unexpected failures may occur.
- When upgrading from early versions of TORQUE (pre-4.0) and Moab, you may encounter a problem where Moab core files are regularly created in `/opt/moab`. This can be caused by old TORQUE library files used by Moab that try to authorize with the old TORQUE `pbs_iff` authorization daemon. You can resolve the problem by removing the old version library files from `/usr/local/lib`.

### To upgrade

1. Build new release (do not install).
2. Stop all TORQUE daemons (see [qterm](#) and [momctl -s](#)).
3. Install new TORQUE (use *make install*).
4. Start all TORQUE daemons.

### Rolling upgrade

If you are upgrading to a new point release of your current version (for example, from 4.2.2 to 4.2.3) and not to a new major release from your current version (for example, from 4.1 to 4.2), you can use the following procedure to upgrade TORQUE without taking your nodes offline.

**i** Because TORQUE version 4.1.4 changed the way that `pbs_server` communicates with the MOMs, it is not recommended that you perform a rolling upgrade of TORQUE from version 4.1.3 to 4.1.4.

## To perform a rolling upgrade in TORQUE

1. Enable the [pbs\\_mom](#) on page 2693 flag on the MOMs you want to upgrade. The `enablemomrestart` option causes a MOM to check if its binary has been updated and restart itself at a safe point when no jobs are running. You can enable this in the MOM configuration file, but it is recommended that you use `momctl` instead.

```
> momctl -q enablemomrestart=1 -h :ALL
```

*The `enablemomrestart` flag is enabled on all nodes.*

2. Replace the `pbs_mom` binary, located in `/usr/local/bin` by default. `pbs_mom` will continue to run uninterrupted because the `pbs_mom` binary has already been loaded in RAM.

```
> torque-package-mom-linux-x86_64.sh --install
```

The next time `pbs_mom` is in an idle state, it will check for changes in the binary. If `pbs_mom` detects that the binary on disk has changed, it will restart automatically, causing the new `pbs_mom` version to load.

After the `pbs_mom` restarts on each node, the `enablemomrestart` parameter will be set back to false (0) for that node.

**i** If you have cluster with high utilization, you may find that the nodes never enter an idle state so `pbs_mom` never restarts. When this occurs, you must manually take the nodes offline and wait for the running jobs to complete before restarting `pbs_mom`. To set the node to an offline state, which will allow running jobs to complete but will not allow any new jobs to be scheduled on that node, use `pbsnodes -o <nodeName>`. After the new MOM has started, you must make the node active again by running `pbsnodes -c <nodeName>`.

## Appendix F: Large cluster considerations

TORQUE has enhanced much of the communication found in the original OpenPBS project. This has resulted in a number of key advantages including support for:

- larger clusters.
- more jobs.
- larger jobs.
- larger messages.

In most cases, enhancements made apply to all systems and no tuning is required. However, some changes have been made configurable to allow site specific modification. The configurable communication parameters are: [node\\_check\\_rate](#), [node\\_ping\\_rate](#), and [tcp\\_timeout](#).

For details, see these topics:

- [Scalability guidelines](#) on page 2812
- [End user command caching](#) on page 2812
- [Moab and TORQUE configuration for large clusters](#) on page 2814
- [Starting TORQUE in large environments](#) on page 2815
- [Other considerations](#) on page 2815

## Scalability guidelines

In very large clusters (in excess of 1,000 nodes), it may be advisable to tune a number of communication layer timeouts. By default, PBS MOM daemons timeout on inter-MOM messages after 60 seconds. In TORQUE 1.1.0p5 and higher, this can be adjusted by setting the timeout parameter in the `mom_priv/config` file (see [Appendix C: Node manager \(MOM\) configuration](#) on page 2783). If 15059 errors (cannot receive message from sisters) are seen in the MOM logs, it may be necessary to increase this value.

Client-to-server communication timeouts are specified via the `tcp_timeout` server option using the `qmgr` command.

**i** On some systems, *ulimit* values may prevent large jobs from running. In particular, the open file descriptor limit (i.e., `ulimit -n`) should be set to at least the maximum job size in procs + 20. Further, there may be value in setting the `fs.file-max` in `sysctl.conf` to a high value, such as:

```
/etc/sysctl.conf:
fs.file-max = 65536
```

### Related topics

- [Appendix F: Large cluster considerations](#) on page 2811

## End user command caching

### qstat

In a large system, users may tend to place excessive load on the system by manual or automated use of resource manager end user client commands. A simple way of reducing this load is through the use of client command wrappers which cache data. The example script below will cache the output of the command `'qstat -f'` for 60 seconds and report this info to end users.

```

#!/bin/sh

# USAGE: qstat $@

CMDPATH=/usr/local/bin/qstat
CACHETIME=60
TMPFILE=/tmp/qstat.f.tmp

if [ "$1" != "-f" ] ; then
    #echo "direct check (arg1=$1) "
    $CMDPATH $1 $2 $3 $4
    exit $?
fi

if [ -n "$2" ] ; then
    #echo "direct check (arg2=$2)"
    $CMDPATH $1 $2 $3 $4
    exit $?
fi

if [ -f $TMPFILE ] ; then
    TMPFILEMTIME=`stat -c %Z $TMPFILE`
else
    TMPFILEMTIME=0
fi

NOW=`date +%s`
AGE=$(( $NOW - $TMPFILEMTIME ))

#echo AGE=$AGE

for i in 1 2 3;do
    if [ "$AGE" -gt $CACHETIME ] ; then
        #echo "cache is stale "

        if [ -f $TMPFILE.1 ] ; then
            #echo someone else is updating cache

            sleep 5

            NOW=`date +%s`

            TMPFILEMTIME=`stat -c %Z $TMPFILE`

            AGE=$(( $NOW - $TMPFILEMTIME ))
        else
            break;
        fi
    fi
done

if [ -f $TMPFILE.1 ] ; then
    #echo someone else is hung

    rm $TMPFILE.1
fi

if [ "$AGE" -gt $CACHETIME ] ; then
    #echo updating cache

    $CMDPATH -f > $TMPFILE.1
mv $TMPFILE.1 $TMPFILE

fi

#echo "using cache"

```

```
cat $TMPFILE
exit 0
```

The above script can easily be modified to cache any command and any combination of arguments by changing one or more of the following attributes:

- script name
- value of \$CMDPATH
- value of \$CACHETIME
- value of \$TMPFILE

For example, to cache the command [pbsnodes](#) -a, make the following changes:

- Move original `pbsnodes` command to `pbsnodes.orig`.
- Save the script as 'pbsnodes'.
- Change \$CMDPATH to `pbsnodes.orig`.
- Change \$TMPFILE to `/tmp/pbsnodes.a.tmp`.

Related topics

- [Appendix F: Large cluster considerations on page 2811](#)

## Moab and TORQUE configuration for large clusters

There are a few basic configurations for Moab and TORQUE that can potentially improve performance on large clusters.

### Moab configuration

In the `moab.cfg` file, add:

1. `RMPOLLINTERVAL 30, 30` - This sets the minimum and maximum poll interval to 30 seconds.
2. `RMCFG[<name>] FLAGS=ASYNCSTART` - This tells Moab not to block until it receives a confirmation that the job starts.
3. `RMCFG[<name>] FLAGS=ASYNCDELETE` - This tells Moab not to block until it receives a confirmation that the job was deleted.

### TORQUE configuration

1. Follow the [Starting TORQUE in large environments](#) recommendations.
2. Increase `job_start_timeout` on `pbs_server`. The default is `300` (5 minutes), but for large clusters the value should be changed to something like `600` (10 minutes). Sites running very large parallel jobs might want to set this value even higher.
3. Use a node health check script on all MOM nodes. This helps prevent jobs from being scheduled on bad nodes and is especially helpful for large parallel jobs.

4. Make sure that `ulimit -n` (maximum file descriptors) is set to *unlimited*, or a very large number, and not the default.
5. For clusters with a high job throughput it is recommended that the server parameter `max_threads` be increased from the default. The default is  $(2 * \text{number of cores} + 1) * 10$ .

#### Related topics

- [Appendix F: Large cluster considerations on page 2811](#)

## Starting TORQUE in large environments

If running TORQUE in a large environment, use these tips to help TORQUE start up faster.

### Fastest possible start up

1. Create a [MOM hierarchy](#), even if your environment has a one-level MOM hierarchy (meaning all MOMs report directly to `pbs_server`), and copy the file to the `mom_priv` directory on the MOMs.
2. Start `pbs_server` with the [-n option](#). This specifies that `pbs_server` won't send the hierarchy to the MOMs unless a MOM requests it.
3. Start the MOMs normally.

### If no daemons are running

1. Start `pbs_server` with the [-c option](#).
2. Start the MOMs without the `-w` option.

### If MOMs are running and just restarting `pbs_server`

1. Start `pbs_server` without the `-c` option.

### If restarting a MOM or all MOMs

1. Start `pbs_server` without the `-w` option. Starting it with `-w` causes the MOMs to appear to be down.

#### Related topics

- [Appendix F: Large cluster considerations on page 2811](#)

## Other considerations

### `job_stat_rate`

In a large system, there may be many users, many jobs, and many requests for information. To speed up response time for users and for programs using the API the [job\\_stat\\_rate](#) can be used to tweak when the `pbs_server` daemon will query MOMs for job information. By increasing this number, a system will not be constantly querying job information and causing other commands to block.

## poll\_jobs

The [poll\\_jobs](#) parameter allows a site to configure how the `pbs_server` daemon will poll for job information. When set to `TRUE`, the `pbs_server` will poll job information in the background and not block on user requests. When set to `FALSE`, the `pbs_server` may block on user requests when it has stale job information data. Large clusters should set this parameter to `TRUE`.

## Internal settings

On large, slow, and/or heavily loaded systems, it may be desirable to increase the `pbs_tcp_timeout` setting used by the `pbs_mom` daemon in MOM-to-MOM communication. This setting defaults to 20 seconds and requires rebuilding code to adjust. For client-server based communication, this attribute can be set using the [qmgr](#) command. For MOM-to-MOM communication, a source code modification is required. To make this change, edit the `$TORQUEBUILDDIR/src/lib/Libifl/tcp_dis.c` file and set `pbs_tcp_timeout` to the desired maximum number of seconds allowed for a MOM-to-MOM request to be serviced.

**i** A system may be heavily loaded if it reports multiple 'End of File from addr' or 'Premature end of message' failures in the `pbs_mom` or `pbs_server` logs.

## Scheduler settings

If using Moab, there are a number of parameters which can be set on the scheduler which may improve TORQUE performance. In an environment containing a large number of short-running jobs, the `JOBAGGREGATIONTIME` parameter (see [Appendix A: Moab Parameters on page 923](#)) can be set to reduce the number of workload and resource queries performed by the scheduler when an event based interface is enabled. If the `pbs_server` daemon is heavily loaded and PBS API timeout errors (i.e. "Premature end of message") are reported within the scheduler, the "TIMEOUT" attribute of the `RMCFG` parameter may be set with a value of between 30 and 90 seconds.

## File system

TORQUE can be configured to disable file system blocking until data is physically written to the disk by using the `--disable-filesync` argument with *configure*. While having `filesync` enabled is more reliable, it may lead to server delays for sites with either a larger number of nodes, or a large number of jobs. `Filesync` is enabled by default.

## Network ARP cache

For networks with more than 512 nodes it is mandatory to increase the kernel's internal ARP cache size. For a network of ~1000 nodes, we use these values in `/etc/sysctl.conf` on all nodes and servers:

```

/etc/sysctl.conf

# Don't allow the arp table to become bigger than this
net.ipv4.neigh.default.gc_thresh3 = 4096
# Tell the gc when to become aggressive with arp table cleaning.
# Adjust this based on size of the LAN.
net.ipv4.neigh.default.gc_thresh2 = 2048
# Adjust where the gc will leave arp table alone
net.ipv4.neigh.default.gc_thresh1 = 1024
# Adjust to arp table gc to clean-up more often
net.ipv4.neigh.default.gc_interval = 3600
# ARP cache entry timeout
net.ipv4.neigh.default.gc_stale_time = 3600

```

Use `sysctl -p` to reload this file.

The ARP cache size on other Unixes can presumably be modified in a similar way.

An alternative approach is to have a static `/etc/ethers` file with all hostnames and MAC addresses and load this by `arp -f /etc/ethers`. However, maintaining this approach is quite cumbersome when nodes get new MAC addresses (due to repairs, for example).

Related topics

- [Appendix F: Large cluster considerations on page 2811](#)

## Appendix G: Prologue and epilogue scripts

TORQUE provides administrators the ability to run scripts before and/or after each job executes. With such a script, a site can prepare systems, perform node health checks, prepend and append text to output and error log files, cleanup systems, and so forth.

The following table shows which MOM runs which script. All scripts must be in the `TORQUE_HOME/mom_priv/` directory and be available on every compute node. The "Mother Superior" is the `pbs_mom` on the first node allocated for a job. While it is technically a sister node, it is not a "Sister" for the purposes of the following table.

 The execution directory for each script is `TORQUE_HOME/mom_priv/`.

Script	Execution location	Execute as	File permissions
<b>prologue</b>	Mother Superior	root	Readable and executable by root and NOT writable by anyone but root (e.g., <code>-r-x-----</code> )
<b>epilogue</b>		root	
<b>prologue.user</b>		user	Readable and executable by root and other (e.g., <code>-r-x--r-x</code> )
<b>epilogue.user</b>		user	

Script	Execution location	Execute as	File permissions
<b>prologue.parallel</b>	Sister	root	Readable and executable by root and NOT writable by anyone but root (e.g., <code>-r-x-----</code> )
<b>epilogue.parallel</b>		root	
<b>prologue.user.parallel</b>		user	Readable and executable by root and other (e.g., <code>-r-x--r-x</code> )
<b>epilogue.user.parallel</b>		user	
<b>epilogue.precancel</b>	Mother Superior This script runs after a job cancel request is received from pbs_server and before a kill signal is sent to the job process.	user	Readable and executable by root and other (e.g., <code>-r-x--r-x</code> )

**i** `epilogue.parallel` is available in version 2.1 and later.

This section contains these topics:

- [Script order of execution on page 2818](#)
- [Script environment on page 2819](#)
- [Per job prologue and epilogue scripts on page 2820](#)
- [Prologue and epilogue scripts time out on page 2821](#)
- [Prologue error processing on page 2821](#)

## Script order of execution

When jobs start, the order of script execution is `prologue` followed by `prologue.user`. On job exit, the order of execution is `epilogue.user` followed by `epilogue` unless a job is canceled. In that case, `epilogue.precancel` is executed first. `epilogue.parallel` is executed only on the Sister nodes when the job is completed.

**i** The `epilogue` and `prologue` scripts are controlled by the system administrator. However, beginning in TORQUE version 2.4 a user `epilogue` and `prologue` script can be used on a per job basis. (See [Per job prologue and epilogue scripts on page 2820](#) for more information.)

**i** Root squashing is now supported for `epilogue` and `prologue` scripts.

Related topics

- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Script environment

The `prologue` and `epilogue` scripts can be very simple. On most systems, the script must declare the execution shell using the `#!<SHELL>` syntax (for example, `#!/bin/sh`). In addition, the script may want to process context sensitive arguments passed by TORQUE to the script.

### Prologue Environment

The following arguments are passed to the `prologue`, `prologue.user`, and `prologue.parallel` scripts:

Argument	Description
<code>argv[1]</code>	job id
<code>argv[2]</code>	job execution user name
<code>argv[3]</code>	job execution group name
<code>argv[4]</code>	job name (TORQUE 1.2.0p4 and higher only)
<code>argv[5]</code>	list of requested resource limits (TORQUE 1.2.0p4 and higher only)
<code>argv[6]</code>	job execution queue (TORQUE 1.2.0p4 and higher only)
<code>argv[7]</code>	job account (TORQUE 1.2.0p4 and higher only)

### Epilogue Environment

TORQUE supplies the following arguments to the `epilogue`, `epilogue.user`, `epilogue.precancel`, and `epilogue.parallel` scripts:

Argument	Description
<code>argv[1]</code>	job id
<code>argv[2]</code>	job execution user name
<code>argv[3]</code>	job execution group name
<code>argv[4]</code>	job name

Argument	Description
<code>argv[5]</code>	session id
<code>argv[6]</code>	list of requested resource limits
<code>argv[7]</code>	list of resources used by job
<code>argv[8]</code>	job execution queue
<code>argv[9]</code>	job account
<code>argv[10]</code>	job exit code

The `epilogue.precancel` script is run after a job cancel request is received by the MOM and before any signals are sent to job processes. If this script exists, it is run whether the canceled job was active or idle.

**i** The cancel job command (`qdel`) will take as long to return as the `epilogue.precancel` script takes to run. For example, if the script runs for 5 minutes, it takes 5 minutes for `qdel` to return.

For all scripts, the environment passed to the script is empty. However, if you submit the job using `msub` rather than `qsub`, some Moab environment variables are available in the TORQUE prologue and epilogue script environment: `MOAB_CLASS`, `MOAB_GROUP`, `MOAB_JOBARRAYINDEX`, `MOAB_JOBARRAYRANGE`, `MOAB_JOBID`, `MOAB_JOBNAME`, `MOAB_MACHINE`, `MOAB_NODECOUNT`, `MOAB_NODELIST`, `MOAB_PARTITION`, `MOAB_PROCCOUNT`, `MOAB_QOS`, `MOAB_TASKMAP`, and `MOAB_USER`. For more information, see [msub -E](#).

Also, standard input for both scripts is connected to a system dependent file. Currently, for all systems this is `/dev/null`. Except for epilogue scripts of an interactive job, `prologue.parallel`, `epilogue.precancel`, and `epilogue.parallel`, the standard output and error are connected to output and error files associated with the job. For an interactive job, since the pseudo terminal connection is released after the job completes, the standard input and error point to `/dev/null`. For `prologue.parallel` and `epilogue.parallel`, the user will need to redirect `stdout` and `stderr` manually.

#### Related topics

- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Per job prologue and epilogue scripts

TORQUE supports per job prologue and epilogue scripts when using the `qsub -l` option. The syntax is:

```
qsub -l prologue=<prologue_script_path> epilogue=<epilogue_script_path>
<script>
```

The path can be either relative (from the directory where the job is submitted) or absolute. The files must be owned by the user with at least execute and read privileges, and the permissions must not be writeable by group or other.

```
/home/usertom/dev/
```

```
-r-x----- 1 usertom usertom 24 2009-11-09 16:11 prologue_script.sh
-r-x----- 1 usertom usertom 24 2009-11-09 16:11 epilogue_script.sh
```

**Example 4-29:**

```
$ qsub -l prologue=/home/usertom/dev/prologue_script.sh,epilogue=/home/usertom/dev/epilogue_script.sh job14.pl
```

This job submission executes the `prologue` script first. When the `prologue` script is complete, `job14.pl` runs. When `job14.pl` completes, the `epilogue` script is executed.

Related topics

- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Prologue and epilogue scripts time out

TORQUE takes preventative measures against prologue and epilogue scripts by placing an alarm around the scripts execution. By default, TORQUE sets the alarm to go off after 5 minutes of execution. If the script exceeds this time, it will be terminated and the node will be marked down. This timeout can be adjusted by setting the `$prologalarm` parameter in the `mom_priv/config` file.

**i** While TORQUE is executing the `epilogue`, `epilogue.user`, or `epilogue.precancel` scripts, the job will be in the *E* (exiting) state.

If an `epilogue.parallel` script cannot open the `.OU` or `.ER` files, an error is logged but the script is continued.

Related topics

- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Prologue error processing

If the `prologue` script executes successfully, it should exit with a zero status. Otherwise, the script should return the appropriate error code as defined in the table below. The `pbs_mom` will report the script's exit status to `pbs_server` which will in turn take the associated action. The following table describes each exit code for the prologue scripts and the action taken.

Error	Description	Action
-4	The script timed out	Job will be requeued

Error	Description	Action
-3	The wait(2) call returned an error	Job will be requeued
-2	Input file could not be opened	Job will be requeued
-1	Permission error (script is not owned by root, or is writable by others)	Job will be requeued
0	Successful completion	Job will run
1	Abort exit code	Job will be aborted
>1	other	Job will be requeued

Example 4-30:

Following are example prologue and epilogue scripts that write the arguments passed to them in the job's standard out file:

prologue	
<b>Script</b>	<pre>#!/bin/sh echo "Prologue Args:" echo "Job ID: \$1" echo "User ID: \$2" echo "Group ID: \$3" echo ""  exit 0</pre>
<b>stdout</b>	<pre>Prologue Args: Job ID: 13724.node01 User ID: user1 Group ID: user1</pre>

epilogue	
<b>Script</b>	<pre>#!/bin/sh echo "Epilogue Args:" echo "Job ID: \$1" echo "User ID: \$2" echo "Group ID: \$3" echo "Job Name: \$4" echo "Session ID: \$5" echo "Resource List: \$6" echo "Resources Used: \$7" echo "Queue Name: \$8" echo "Account String: \$9" echo ""  exit 0</pre>
<b>stdout</b>	<pre>Epilogue Args: Job ID: 13724.node01 User ID: user1 Group ID: user1 Job Name: script.sh Session ID: 28244 Resource List: neednodes=node01,nodes=1,walltime=00:01:00 Resources Used: cput=00:00:00,mem=0kb,vmem=0kb,walltime=00:00:07 Queue Name: batch Account String:</pre>

**Example 4-31:**

The Ohio Supercomputer Center contributed the following scripts:

"prologue creates a unique temporary directory on each node assigned to a job before the job begins to run, and epilogue deletes that directory after the job completes.



Having a separate temporary directory on each node is probably not as good as having a good, high performance parallel filesystem.

```

prologue

#!/bin/sh
# Create TMPDIR on all the nodes
# Copyright 1999, 2000, 2001 Ohio Supercomputer Center
# prologue gets 3 arguments:
# 1 -- jobid
# 2 -- userid
# 3 -- grpuid
#
jobid=$1
user=$2
group=$3
nodefile=/var/spool/pbs/aux/$jobid
if [ -r $nodefile ] ; then
    nodes=$(sort $nodefile | uniq)
else
    nodes=localhost
fi
tmp=/tmp/pbstmp.$jobid
for i in $nodes ; do
    ssh $i mkdir -m 700 $tmp \&\& chown $user.$group $tmp
done
exit 0

```

```

epilogue

#!/bin/sh
# Clear out TMPDIR
# Copyright 1999, 2000, 2001 Ohio Supercomputer Center
# epilogue gets 9 arguments:
# 1 -- jobid
# 2 -- userid
# 3 -- grpuid
# 4 -- job name
# 5 -- sessionid
# 6 -- resource limits
# 7 -- resources used
# 8 -- queue
# 9 -- account
#
jobid=$1
nodefile=/var/spool/pbs/aux/$jobid
if [ -r $nodefile ] ; then
    nodes=$(sort $nodefile | uniq)
else
    nodes=localhost
fi
tmp=/tmp/pbstmp.$jobid
for i in $nodes ; do
    ssh $i rm -rf $tmp
done
exit 0

```

**i** prologue, prologue.user, and prologue.parallel scripts can have dramatic effects on job scheduling if written improperly.

### Related topics

- [Appendix G: Prologue and epilogue scripts on page 2817](#)

## Appendix H: Running multiple TORQUE servers and MOMs on the same node

TORQUE can be configured to allow multiple servers and MOMs to run on the same node. This example will show how to configure, compile and install two different TORQUE servers and MOMs on the same node. For details, see these topics:

- [Configuring the first TORQUE on page 2825](#)
- [Configuring the second TORQUE on page 2825](#)
- [Bringing the first TORQUE server online on page 2825](#)
- [Bringing the second TORQUE server online on page 2826](#)

### Configuring the first TORQUE

```
./configure --with-server-home=/usr/spool/PBS1 --bindir=/usr/spool/PBS1/bin --
sbindir=/usr/spool/PBS1/sbin
```

Then `make` and `make install` will place the first TORQUE into `/usr/spool/PBS1` with the executables in their corresponding directories.

### Configuring the second TORQUE

```
./configure --with-server-home=/usr/spool/PBS2 --bindir=/usr/spool/PBS2/bin --
sbindir=/usr/spool/PBS2/sbin
```

Then `make` and `make install` will place the second TORQUE into `/usr/spool/PBS2` with the executables in their corresponding directories.

### Bringing the first TORQUE server online

Each command, including `pbs_server` and `pbs_mom`, takes parameters indicating which servers and ports to connect to or listen on (when appropriate). Each of these is documented in their corresponding man pages (configure with `--enable-docs`).

In this example the first TORQUE server will accept batch requests on port 35000, communicate with the MOMs on port 35001, and communicate via RPP on port 35002. The first TORQUE MOM will try to connect to the server on port 35000, it will listen for requests from the server on port 35001 and will communicate via RPP on port 35002. (Each of these command arguments is discussed in further details on the corresponding man page. In particular, `-t create` is only used the first time a server is run.)

```
> pbs_server -p 35000 -M 35001 -R 35002 -t create
> pbs_mom -S 35000 -M 35001 -R 35002
```

Afterwards, when using a client command to make a batch request it is necessary to specify the server name and server port (35000):

```
> pbsnodes -a -s node01:35000
```

Submitting jobs can be accomplished using the `-q` option (`[[queue]][@host[:port]]`):

```
> qsub -q @node01:35000 /tmp/script.pbs
```

## Bringing the second TORQUE server online

In this example the second TORQUE server will accept batch requests on port 36000, communicate with the MOMS on port 36002, and communicate via RPP on port 36002. The second TORQUE MOM will try to connect to the server on port 36000, it will listen for requests from the server on port 36001 and will communicate via RPP on port 36002.

```
> pbs_server -p 36000 -M 36001 -R 36002 -t create
> pbs_mom -S 36000 -M 36001 -R 36002
```

Afterward, when using a client command to make a batch request it is necessary to specify the server name and server port (36002):

```
> pbsnodes -a -s node01:36000
> qsub -q @node01:36000 /tmp/script.pbs
```

## Appendix I: Security overview

The authorization model for TORQUE changed in version 4.0.0 from `pbs_iff` to a daemon called `trqauthd`. The job of the `trqauthd` daemon is the same as `pbs_iff`. The difference is that `trqauthd` is a resident daemon whereas `pbs_iff` is invoked by each client command. `pbs_iff` is not scalable and is prone to failure under even small loads. `trqauthd` is very scalable and creates the possibility for better security measures in the future.

### trqauthd and pbs\_iff authorization theory

The key to security of both `trqauthd` and `pbs_iff` is the assumption that any host which has been added to the TORQUE cluster has been secured by the administrator. Neither `trqauthd` nor `pbs_iff` do authentication. They only do authorization of users. Given that the host system is secure the following is the procedure by which `trqauthd` and `pbs_iff` authorize users to `pbs_server`.

1. Client utility makes a connection to `pbs_server` on a dynamic port.
2. Client utility sends a request to `trqauthd` with the user name and port.
3. `trqauthd` verifies the user ID and then sends a request to `pbs_server` on a privileged port with the user ID and dynamic port to authorize the connection.
4. `trqauthd` reports results of the server to client utility.

Both `trqauthd` and `pbs_iff` use Unix domain sockets for communication from the client utility. Unix domain sockets have the ability to verify that a user is who they say they are by using security features that are part of the file system.

## Appendix J: Job submission filter ("qsub wrapper")

When a "submit filter" exists, TORQUE will send the command file (or contents of STDIN if piped to `qsub`) to that script/executable and allow it to evaluate the submitted request based on specific site policies. The resulting file is then handed back to `qsub` and processing continues. Submit filters can check user jobs for correctness based on site policies. They can also modify user jobs as they are submitted. Some examples of what a submit filter might evaluate and check for are:

- Memory Request - Verify that the job requests memory and rejects if it does not.
- Job event notifications - Check if the job does one of the following and rejects it if it:
  - explicitly requests no notification.
  - requests notifications but does not provide an email address.
- Walltime specified - Verify that the walltime is specified.
- Global Walltime Limit - Verify that the walltime is below the global max walltime.
- Test Walltime Limit - If the job is a test job, this check rejects the job if it requests a walltime longer than the testing maximum.

The script below reads the original submission request from STDIN and shows how you could insert parameters into a job submit request:

```
#!/bin/sh
# add default memory constraints to all requests
# that did not specify it in user's script or on command line
echo "#PBS -l mem=16MB"
while read i
do
echo $i
done
```

**i** If you use a `qsub` script that includes `#PBS` directives to pass arguments instead of on the command line; for example,

```
#!/bin/sh
#
#This is an example script example.sh
#
#These commands set up the Grid Environment for your job:
#PBS -N ExampleJob
#PBS -l nodes=1,walltime=00:01:00
#PBS -q np_workq
#PBS -M YOURUNIQNAME@umich.edu
#PBS -m abe

#print the time and date
date

#wait 10 seconds
sleep 10

#print the time and date again
date
```

then your submit filter script must detect these directives and print them to stdout in order for `qsub` to see them.,

The same command line arguments passed to `qsub` will be passed to the submit filter and in the same order. Exit status of `-1` will cause `qsub` to reject the submission with a message stating that it failed due to administrative policies.

The "submit filter" must be executable, must be available on each of the nodes where users may submit jobs, and by default must be located at `${libexecdir}/qsub_filter` (for version 2.1 and older: `/usr/local/sbin/torque_submitfilter`). At run time, if the file does not exist at this new preferred path then `qsub` will fall back to the old hard-coded path. The submit filter location can be customized by setting the `SUBMITFILTER` parameter inside the file (see [Appendix K: "torque.cfg" configuration file on page 2828](#)), as in the following example:

torque.cfg:

```
SUBMITFILTER /opt/torque/submit.pl
...
```

**i** Initial development courtesy of Oak Ridge National Laboratories.

## Appendix K: "torque.cfg" configuration file

### CLIENTRETRY

<b>Format</b>	<INT>
---------------	-------

CLIENTRETRY	
<b>Default</b>	0
<b>Description</b>	Seconds between retry attempts to talk to pbs_server.
<b>Example</b>	<pre>CLIENTRETRY 10</pre> <p><i>TORQUE waits 10 seconds after a failed attempt before it attempts to talk to pbs_server again.</i></p>

DEFAULTCKPT	
<b>For mat</b>	One of <i>None, Enabled, Shutdown, Periodic, Interval=minutes, depth=number, or dir=path</i>
<b>Default</b>	<i>None</i>
<b>Description</b>	Default value for job's checkpoint attribute. For a description of all possible values, see <a href="#">qsub on page 2744</a> . <div style="border: 1px solid #005596; border-radius: 5px; padding: 5px; margin-top: 5px;">  This default setting can be overridden at job submission with the <code>qsub -c</code> option.           </div>
<b>Example</b>	<pre>DEFAULTCKPT Shutdown</pre> <p><i>By default, TORQUE checkpoints at pbs_mom shutdown.</i></p>

FAULT_TOLERANT_BY_DEFAULT	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Sets all jobs to fault tolerant by default. (See <a href="#">qsub -f</a> for more information on fault tolerance.)
<b>Example</b>	<pre>FAULT_TOLERANT_BY_DEFAULT TRUE</pre> <p><i>Jobs are fault tolerant by default. They will not be canceled based on failed polling, no matter how many nodes fail to report.</i></p>

HOST_NAME_SUFFIX	
<b>Format</b>	<STRING>
<b>Default</b>	---
<b>Description</b>	Specifies a hostname suffix. When <code>qsub</code> submits a job, it also submits the username of the submitter and the name of the host from which the user submitted the job. TORQUE appends the value of <code>HOST_NAME_SUFFIX</code> to the hostname. This is useful for multi-homed systems that may have more than one name for a host.
<b>Example</b>	<pre>HOST_NAME_SUFFIX -ib</pre> <p><i>When a job is submitted, the -ib suffix is appended to the host name.</i></p>

QSUBHOST	
<b>Format</b>	<HOSTNAME>
<b>Default</b>	---
<b>Description</b>	The hostname given as the argument of this option will be used as the <code>PBS_O_HOST</code> variable for job submissions. By default, <code>PBS_O_HOST</code> is the hostname of the submission host. This option allows administrators to override the default hostname and substitute a new name.
<b>Example</b>	<pre>QSUBHOST host1</pre> <p><i>The default hostname associated with a job is host1.</i></p>

QSUBSENDUID	
<b>Format</b>	N/A
<b>Default</b>	---
<b>Description</b>	Integer for job's <code>PBS_OUID</code> variable. Specifying the parameter name anywhere in the config file enables the feature. Removing the parameter name disables the feature.
<b>Example</b>	<pre>QSUBSENDUID</pre> <p><i>TORQUE assigns a unique ID to a job when it is submitted by <code>qsub</code>.</i></p>

QSUBSLEEP	
<b>Format</b>	<INT>
<b>Default</b>	0
<b>Description</b>	Specifies time, in seconds, to sleep between a user's submitting and TORQUE's starting a <code>qsub</code> command. Used to prevent users from overwhelming the scheduler.
<b>Example</b>	<pre>QSUBSLEEP 2</pre> <p><i>When a job is submitted with <code>qsub</code>, it will sleep for 2 seconds.</i></p>

RERUNNABLEBYDEFAULT	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Specifies if a job is re-runnable by default. Setting this to false causes the re-runnable attribute value to be false unless the users specifies otherwise with the <code>qsub -r</code> option. (New in TORQUE 2.4.)
<b>Example</b>	<pre>RERUNNABLEBYDEFAULT FALSE</pre> <p><i>By default, <code>qsub</code> jobs cannot be rerun.</i></p>

SERVERHOST	
<b>Format</b>	<STRING>
<b>Default</b>	localhost
<b>Description</b>	If set, the <a href="#">qsub on page 2744</a> command will open a connection to the host specified by the SERVERHOST string.
<b>Example</b>	<pre>SERVERHOST orion15</pre> <p><i>The server will open socket connections and and communicate using serverhost <code>orion15</code>.</i></p>

SUBMITFILTER	
<b>Format</b>	<STRING>
<b>Default</b>	`\${libexecdir}/qsub_filter (for version 2.1 and older: /usr/local/sbin/torque_submitfilter)
<b>Description</b>	Specifies the location of the submit filter (see <a href="#">Appendix J: Job submission filter ("qsub wrapper") on page 2827</a> used to pre-process job submission.
<b>Example</b>	<pre>SUBMITFILTER /usr/local/sbin/qsub_filter</pre> <p><i>The location of the submit filter is specified as /usr/local/sbin/qsub_filter.</i></p>

TRQ_IFNAME	
<b>Format</b>	<STRING>
<b>Default</b>	null
<b>Description</b>	Allows you to specify a specific network interface to use for outbound TORQUE requests. The string is the name of a network interface, such as <i>eth0</i> or <i>eth1</i> , depending on which interface you want to use.
<b>Example</b>	<pre>TRQ_IFNAME eth1</pre> <p><i>Outbound TORQUE requests are handled by eth1.</i></p>

VALIDATEGROUP	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	FALSE
<b>Description</b>	Validate submit user's group on <a href="#">qsub</a> commands. For TORQUE builds released after 2/8/2011, <i>VALIDATEGROUP</i> also checks any groups requested in <i>group_list</i> at the submit host. Set <i>VALIDATEGROUP</i> to "TRUE" if you set <a href="#">disable_server_id_check</a> to TRUE.
<b>Example</b>	<pre>VALIDATEGROUP TRUE</pre> <p><i>qsub verifies the submitter's group ID.</i></p>

VALIDATEPATH	
<b>Format</b>	<BOOLEAN>
<b>Default</b>	TRUE
<b>Description</b>	Validate local existence of '-d' working directory.
<b>Example</b>	<pre>VALIDATEPATH FALSE</pre> <p><i>qsub does not validate the path.</i></p>

## Appendix L: TORQUE Quick Start Guide

### Initial installation

TORQUE is now hosted at <https://github.com> under the adaptivecomputing organization. To download source, you will need to use the [git](#) utility. For example:

```
[root]# git clone https://github.com/adaptivecomputing.com/torque.git -b 5.0.0 5.0.0
```

To download a different version, replace each 5.0.0 with the desired version. After downloading a copy of the repository, you can list the current branches by typing `git branch -a` from within the directory of the branch you cloned.

**i** If you're checking source out from git, read the `README.building-40` file in the repository.

Extract and build the distribution on the machine that will act as the "TORQUE server" - the machine that will monitor and control all compute nodes by running the `pbs_server` daemon. See the example below:

```
> tar -xzf torque.tar.gz
> cd torque
> ./configure
> make
> make install
```

**i** OSX 10.4 users need to change the `#define _TDARWIN` in `src/include/pbs_config.h` to `#define _TDARWIN_8`.

**i** After installation, verify you have PATH environment variables configured for `/usr/local/bin/` and `/usr/local/sbin/`. Client commands are installed to `/usr/local/bin` and server binaries are installed to `/usr/local/sbin`.

**i** In this document, TORQUE\_HOME corresponds to where TORQUE stores its configuration files. The default is `/var/spool/torque`.

## Initialize/Configure TORQUE on the server (pbs\_server)

- Once installation on the TORQUE server is complete, configure the pbs\_server daemon by executing the command `torque.setup <USER>` found packaged with the distribution source code, where `<USER>` is a username that will act as the TORQUE admin. This script will set up a basic batch queue to get you started. If you experience problems, make sure that the most recent TORQUE executables are being executed, or that the executables are in your current PATH.

**i** If you are upgrading from TORQUE 2.5.9, run `pbs_server -u` before running `torque.setup`.

```
[root]# pbs_server -u
```

- If doing this step manually, be certain to run the command `pbs_server -t create` to create the new batch database. If this step is not taken, the pbs\_server daemon will be unable to start.
- Proper server configuration can be verified by following the steps listed in Testing server configuration.

## Install TORQUE on the compute nodes

To configure a compute node do the following on each machine (see page 19, Section 3.2.1 of *PBS Administrators Manual* for full details):

- Create the self-extracting, distributable packages with `make packages` (See the `INSTALL` file for additional options and features of the distributable packages) and use the parallel shell command from your cluster management suite to copy and execute the package on all nodes (i.e. xCAT users might do `prcp torque-package-linux-i686.sh main:/tmp/; psh main /tmp/torque-package-linux-i686.sh --install`). Optionally, distribute and install the clients package.

## Configure TORQUE on the compute nodes

- For each compute host, the MOM daemon must be configured to trust the pbs\_server daemon. In TORQUE 2.0.0p4 and earlier, this is done by creating the `TORQUE_HOME/mom_priv/config` file and setting the `$pbsserver` parameter. In TORQUE 2.0.0p5 and later, this can also be done by creating the `TORQUE_HOME/server_name` file and placing the server hostname inside.
- Additional config parameters may be added to `TORQUE_HOME/mom_priv/config` (see [Appendix C: Node manager \(MOM\) configuration on page 2783](#) for details).

## Configure data management on the compute nodes

Data management allows jobs' data to be staged in/out or to and from the server and compute nodes.

- For shared filesystems (i.e., NFS, DFS, AFS, etc.) use the [\\$usecp](#) parameter in the `mom_priv/config` files to specify how to map a user's home directory.  
(Example: `$usecp gridmaster.tmx.com:/home /home`)
- For local, non-shared filesystems, `rcp` or `scp` must be configured to allow direct copy without prompting for passwords (key authentication, etc.)

## Update TORQUE server configuration

On the TORQUE server, append the list of newly configured compute nodes to the `TORQUE_HOME/server_priv/nodes` file:

```
server_priv/nodes
computenode001.cluster.org
computenode002.cluster.org
computenode003.cluster.org
```

## Start the pbs\_mom daemons on compute nodes

- Next start the `pbs_mom` daemon on each compute node by running the `pbs_mom` executable.

Run the `trqauthd` daemon to run client commands (see [Configuring trqauthd for client commands on page 2556](#)). This enables running client commands.

## Verifying correct TORQUE installation

The `pbs_server` daemon was started on the TORQUE server when the `torque.setup` file was executed or when it was manually configured. It must now be restarted so it can reload the updated configuration changes.

```
# shutdown server
> qterm # shutdown server

# start server
> pbs_server

# verify all queues are properly configured
> qstat -q

# view additional server configuration
> qmgr -c 'p s'

# verify all nodes are correctly reporting
> pbsnodes -a

# submit a basic job
> echo "sleep 30" | qsub

# verify jobs display
> qstat
```

At this point, the job will not start because there is no scheduler running. The scheduler is enabled in the next step below.

## Enabling the scheduler

Selecting the cluster scheduler is an important decision and significantly affects cluster utilization, responsiveness, availability, and intelligence. The default TORQUE scheduler, `pbs_sched`, is very basic and will provide poor utilization of your cluster's resources. Other options, such as [Maui Scheduler](#) or [Moab Workload Manager](#) are highly recommended. If using Maui/Moab, refer to the [Moab-TORQUE Integration Guide on page 1228](#). If using `pbs_sched`, start this daemon now.

 If you are installing ClusterSuite, TORQUE and Moab were configured at installation for interoperability and no further action is required.

## Startup/Shutdown service script for TORQUE/Moab (OPTIONAL)

Optional startup/shutdown service scripts are provided as an example of how to run TORQUE as an OS service that starts at bootup. The scripts are located in the `contrib/init.d/` directory of the TORQUE tarball you downloaded. In order to use the script you must:

- Determine which `init.d` script suits your platform the best.
- Modify the script to point to TORQUE's install location. This should only be necessary if you used a non-default install location for TORQUE (by using the `--prefix` option of `./configure`).
- Place the script in the `/etc/init.d/` directory.
- Use a tool like `chkconfig` to activate the start-up scripts or make symbolic links (`S99moab` and `K15moab`, for example) in desired runtimes (`/etc/rc.d/rc3.d/` on Redhat, etc.).

Related topics

- [Advanced configuration on page 2558](#)

## Appendix M: BLCR acceptance tests

This section contains a description of the testing done to verify the functionality of the BLCR implementation. For details, see these topics:

- [Test environment on page 2836](#)
- [Test 1 - Basic operation on page 2837](#)
- [Test 2 - Persistence of checkpoint images on page 2839](#)
- [Test 3 - Restart after checkpoint on page 2840](#)
- [Test 4 - Multiple checkpoint/restart on page 2841](#)
- [Test 5 - Periodic checkpoint on page 2841](#)
- [Test 6 - Restart from previous image on page 2842](#)

### Test environment

All these tests assume the following test program and shell script, `test.sh`.

```
#include
int main( int argc, char *argv[] )
{
  int i;

  for (i=0; i<100; i++)
  {
    printf("i = %d\n", i);
    fflush(stdout);
    sleep(1);
  }
}
#!/bin/bash

/home/test/test
```

### Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 1 - Basic operation

### Introduction

This test determines if the proper environment has been established.

### Test steps

Submit a test job and the issue a hold on the job.

```
> qsub -c enabled test.sh
999.xxx.yyy
> qhold 999
```

### Possible failures

Normally the result of `qhold` is nothing. If an error message is produced saying that `qhold` is not a supported feature then one of the following configuration errors might be present.

- The TORQUE images may have not be configured with `--enable-blcr`
- BLCR support may not be installed into the kernel with `insmod`.
- The config script in `mom_priv` may not exist with `$checkpoint_script` defined.
- The config script in `mom_priv` may not exist with `$restart_script` defined.
- The config script in `mom_priv` may not exist with `$checkpoint_run_exe` defined.
- The scripts referenced in the config file may not exist.
- The scripts referenced in the config file may not have the correct permissions.

## Successful results

If no configuration was done to specify a specific directory location for the checkpoint file, the default location is off of the TORQUE directory, which in my case is `/var/spool/torque/checkpoint`.

Otherwise, go to the specified directory for the checkpoint image files. This was done by either specifying an option on job submission, i.e. `-c dir=/home/test` or by setting an attribute on the execution queue. This is done with the command `qmgr -c 'set queue batch checkpoint_dir=/home/test'`.

Doing a directory listing shows the following.

```
# find /var/spool/torque/checkpoint
/var/spool/torque/checkpoint
/var/spool/torque/checkpoint/999.xxx.yyy.CK
/var/spool/torque/checkpoint/999.xxx.yyy.CK/ckpt.999.xxx.yyy.1205266630
# find /var/spool/torque/checkpoint |xargs ls -l
-r----- 1 root root 543779 2008-03-11 14:17
/var/spool/torque/checkpoint/999.xxx.yyy.CK/ckpt.999.xxx.yyy.1205266630

/var/spool/torque/checkpoint:
total 4
drwxr-xr-x 2 root root 4096 2008-03-11 14:17 999.xxx.yyy.CK

/var/spool/torque/checkpoint/999.xxx.yyy.CK:
total 536
-r----- 1 root root 543779 2008-03-11 14:17 ckpt.999.xxx.yyy.1205266630
```

Doing a `qstat -f` command should show the job in a held state, `job_state = H`. Note that the attribute `checkpoint_name` is set to the name of the file seen above.

If a checkpoint directory has been specified, there will also be an attribute `checkpoint_dir` in the output of `qstat -f`.

```

$ qstat -f
Job Id: 999.xxx.yyy
Job_Name = test.sh
Job_Owner = test@xxx.yyy
resources_used.cput = 00:00:00
resources_used.mem = 0kb
resources_used.vmem = 0kb
resources_used.walltime = 00:00:06
job_state = H
queue = batch
server = xxx.yyy
Checkpoint = u
ctime = Tue Mar 11 14:17:04 2008
Error_Path = xxx.yyy:/home/test/test.sh.e999
exec_host = test/0
Hold_Types = u
Join_Path = n
Keep_Files = n
Mail_Points = a
mtime = Tue Mar 11 14:17:10 2008
Output_Path = xxx.yyy:/home/test/test.sh.o999
Priority = 0
qtime = Tue Mar 11 14:17:04 2008
Rerunable = True
Resource_List.needsnodes = 1
Resource_List.nodect = 1
Resource_List.nodes = 1
Resource_List.walltime = 01:00:00
session_id = 9402 substate = 20
Variable_List = PBS_O_HOME=/home/test,PBS_O_LANG=en_US.UTF-8,
                PBS_O_LOGNAME=test,
                PBS_O_PATH=/usr/local/perltests/bin:/home/test/bin:/usr/local/s
bin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games,
                PBS_O_SHELL=/bin/bash,PBS_SERVER=xxx.yyy,
                PBS_O_HOST=xxx.yyy,PBS_O_WORKDIR=/home/test,
                PBS_O_QUEUE=batch
euser = test
egroup = test
hashname = 999.xxx.yyy
queue_rank = 3
queue_type = E comment = Job started on Tue Mar 11 at 14:17
exit_status = 271
submit_args = test.sh
start_time = Tue Mar 11 14:17:04 2008
start_count = 1
checkpoint_dir = /var/spool/torque/checkpoint/999.xxx.yyy.CK
checkpoint_name = ckpt.999.xxx.yyy.1205266630

```

 The value of Resource\_List.\* is the amount of resources requested.

### Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 2 - Persistence of checkpoint images

### Introduction

This test determines if the checkpoint files remain in the default directory after the job is removed from the TORQUE queue.

Note that this behavior was requested by a customer but in fact may not be the right thing to do as it leaves the checkpoint files on the execution node. These will gradually build up over time on the node being limited only by disk space. The right thing would seem to be that the checkpoint files are copied to the user's home directory after the job is purged from the execution node.

## Test steps

Assuming the steps of Test 1 (see [Test 1 - Basic operation on page 2837](#)), delete the job and then wait until the job leaves the queue after the completed job hold time. Then look at the contents of the default checkpoint directory to see if the files are still there.

```
> qsub -c enabled test.sh
999.xxx.yyy
> qhold 999
> qdel 999
> sleep 100
> qstat
>
> find /var/spool/torque/checkpoint
... files ...
```

## Possible failures

The files are not there, did Test 1 actually pass?

## Successful results

The files are there.

Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 3 - Restart after checkpoint

### Introduction

This test determines if the job can be restarted after a checkpoint hold.

### Test steps

Assuming the steps of Test 1 (see [Test 1 - Basic operation on page 2837](#)), issue a `qrls` command. Have another window open into the `/var/spool/torque/spool` directory and tail the job.

### Successful results

After the `qrls`, the job's output should resume.

Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 4 - Multiple checkpoint/restart

### Introduction

This test determines if the checkpoint/restart cycle can be repeated multiple times.

### Test steps

Start a job and then while tailing the job output, do multiple [qhold/qrls](#) operations.

```
> qsub -c enabled test.sh
999.xxx.yyy
> qhold 999
> qrls 999
> qhold 999
> qrls 999
> qhold 999
> qrls 999
```

#### Successful results

After each `qrls`, the job's output should resume. Also tried "while true; do `qrls 999`; `qhold 999`; done" and this seemed to work as well.

#### Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 5 - Periodic checkpoint

### Introduction

This test determines if automatic periodic checkpoint will work.

### Test steps

Start the job with the option `-c enabled,periodic,interval=1` and look in the checkpoint directory for checkpoint images to be generated about every minute.

```
> qsub -c enabled,periodic,interval=1 test.sh
999.xxx.yyy
```

#### Successful results

After each `qrls`, the job's output should resume. Also tried "while true; do `qrls 999`; `qhold 999`; done" and this seemed to work as well.

#### Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)

## Test 6 - Restart from previous image

### Introduction

This test determines if the job can be restarted from a previous checkpoint image.

### Test steps

Start the job with the option `-c enabled,periodic,interval=1` and look in the checkpoint directory for checkpoint images to be generated about every minute. Do a [qhold](#) on the job to stop it. Change the attribute `checkpoint_name` with the [qalter](#) command. Then do a [qrls](#) to restart the job.

```
> qsub -c enabled,periodic,interval=1 test.sh
999.xxx.yyy
> qhold 999
> qalter -W checkpoint_name=ckpt.999.xxx.yyy.1234567
> qrls 999
```

### Successful results

The job output file should be truncated back and the count should resume at an earlier number.

### Related topics

- [Appendix M: BLCR acceptance tests on page 2836](#)