

Bright Cluster Manager 7.3

# Developer Manual

Revision: 67c5798

Date: Mon Dec 10 2018



©2017 Bright Computing, Inc. All Rights Reserved. This manual or parts thereof may not be reproduced in any form unless permitted by contract or by written permission of Bright Computing, Inc.

## **Trademarks**

Linux is a registered trademark of Linus Torvalds. PathScale is a registered trademark of Cray, Inc. Red Hat and all Red Hat-based trademarks are trademarks or registered trademarks of Red Hat, Inc. SUSE is a registered trademark of Novell, Inc. PGI is a registered trademark of NVIDIA Corporation. FLEXlm is a registered trademark of Flexera Software, Inc. ScaleMP is a registered trademark of ScaleMP, Inc. All other trademarks are the property of their respective owners.

## **Rights and Restrictions**

All statements, specifications, recommendations, and technical information contained herein are current or planned as of the date of publication of this document. They are reliable as of the time of this writing and are presented without warranty of any kind, expressed or implied. Bright Computing, Inc. shall not be liable for technical or editorial errors or omissions which may occur in this document. Bright Computing, Inc. shall not be liable for any damages resulting from the use of this document.

## **Limitation of Liability and Damages Pertaining to Bright Computing, Inc.**

The Bright Cluster Manager product principally consists of free software that is licensed by the Linux authors free of charge. Bright Computing, Inc. shall have no liability nor will Bright Computing, Inc. provide any warranty for the Bright Cluster Manager to the extent that is permitted by law. Unless confirmed in writing, the Linux authors and/or third parties provide the program as is without any warranty, either expressed or implied, including, but not limited to, marketability or suitability for a specific purpose. The user of the Bright Cluster Manager product shall accept the full risk for the quality or performance of the product. Should the product malfunction, the costs for repair, service, or correction will be borne by the user of the Bright Cluster Manager product. No copyright owner or third party who has modified or distributed the program as permitted in this license shall be held liable for damages, including general or specific damages, damages caused by side effects or consequential damages, resulting from the use of the program or the un-usability of the program (including, but not limited to, loss of data, incorrect processing of data, losses that must be borne by you or others, or the inability of the program to work together with any other program), even if a copyright owner or third party had been advised about the possibility of such damages unless such copyright owner or third party has signed a writing to the contrary.

# Table of Contents

Table of Contents . . . . .	i
0.1 About This Manual . . . . .	xv
0.2 About The Manuals In General . . . . .	xv
0.3 Getting Administrator-Level Support . . . . .	xvi
0.4 Getting Developer-Level Support . . . . .	xvi
0.5 Getting Professional Services . . . . .	xvi
<b>1 Bright Cluster Manager Python API</b>	<b>1</b>
1.1 Installation . . . . .	1
1.1.1 Windows Clients . . . . .	1
1.1.2 Linux Clients . . . . .	2
1.2 Examples . . . . .	2
1.2.1 First Program . . . . .	2
1.3 Methods And Properties . . . . .	3
1.3.1 Viewing All Properties And Methods . . . . .	3
1.3.2 Property Lists . . . . .	4
1.3.3 Creating New Objects . . . . .	4
1.3.4 List Of Objects . . . . .	5
1.3.5 Useful Methods . . . . .	7
1.3.6 Useful Example Program . . . . .	7
1.4 The Workload Management API . . . . .	9
1.4.1 Job Submission . . . . .	9
1.4.2 Job Information And Management . . . . .	12
1.4.3 Queue Information And Management . . . . .	13
<b>2 Metric Collections</b>	<b>15</b>
2.1 Metric Collections Added Using <code>cmsh</code> . . . . .	15
2.2 Metric Collections Initialization . . . . .	15
2.3 Metric Collections Output During Regular Use . . . . .	17
2.4 Metric Collections Error Handling . . . . .	17
2.5 Metric Collections Consolidator Syntax . . . . .	17
2.6 Metric Collections Environment Variables . . . . .	18
2.7 Metric Collections Examples . . . . .	20
2.8 Metric Collections On iDataPlex And Similar Units . . . . .	20
<b>3 Bright Cluster Manager JSON API</b>	<b>23</b>
3.1 Services . . . . .	23
3.1.1 <code>auth</code> . . . . .	23
3.1.2 <code>ceph</code> . . . . .	23
3.1.3 <code>cert</code> . . . . .	23
3.1.4 <code>cloud</code> . . . . .	23

3.1.5	device	23
3.1.6	gui	23
3.1.7	hadoop	23
3.1.8	job	23
3.1.9	keyvalue	23
3.1.10	lustre	23
3.1.11	main	23
3.1.12	mon	23
3.1.13	net	23
3.1.14	openstack	23
3.1.15	part	23
3.1.16	proc	23
3.1.17	prov	23
3.1.18	puppet	23
3.1.19	serv	23
3.1.20	session	23
3.1.21	test	23
3.1.22	ticket	23
3.1.23	user	23
3.2	Entities	23
3.2.1	BackupRole	23
3.2.2	BadEntityManagers	24
3.2.3	BasicResource	24
3.2.4	BillingHistory	24
3.2.5	BootRole	24
3.2.6	BurnConfig	24
3.2.7	BurnStatus	24
3.2.8	BurnTestStatus	24
3.2.9	Category	24
3.2.10	Ceph	24
3.2.11	CephMonitorRole	24
3.2.12	CephOSDAssociation	24
3.2.13	CephOSDPool	24
3.2.14	CephOSDRole	24
3.2.15	CephState	24
3.2.16	Certificate	24
3.2.17	CertificateRequest	24
3.2.18	Chassis	24
3.2.19	CloudDirectorRole	24
3.2.20	CloudGatewayRole	24
3.2.21	CloudImage	24
3.2.22	CloudJobDescription	24
3.2.23	CloudJobSubmissionStatus	24
3.2.24	CloudNode	24
3.2.25	CloudPrivateCloud	24
3.2.26	CloudProvider	24

---

3.2.27	CloudRegion	24
3.2.28	CloudSettings	24
3.2.29	CloudStaticIP	24
3.2.30	CloudStorageAction	24
3.2.31	CloudStorageNodeState	24
3.2.32	CloudType	24
3.2.33	CloudVirtualNetworkInterface	24
3.2.34	ClusterSetup	24
3.2.35	CMDaemonBackgroundTask	24
3.2.36	CMDaemonFailover	24
3.2.37	CMDaemonFailoverGroup	24
3.2.38	CMDaemonFailoverGroupStatus	24
3.2.39	CMDaemonFailoverPeer	24
3.2.40	CMDaemonFailoverStatus	24
3.2.41	CMDaemonStatus	24
3.2.42	CMService	24
3.2.43	CondorClientRole	24
3.2.44	CondorJob	24
3.2.45	CondorJobQueue	24
3.2.46	CondorJobQueueStat	24
3.2.47	CondorServerRole	24
3.2.48	ConfigSum	24
3.2.49	ConfigurationOverlay	25
3.2.50	ConsolidatorConf	25
3.2.51	DatabaseRole	25
3.2.52	DellClustat	25
3.2.53	DellClustatGroup	25
3.2.54	DellClustatNode	25
3.2.55	DellDiskGroupInfo	25
3.2.56	DellPhysicalDiskDriveInfo	25
3.2.57	DellRAIDControllerInfo	25
3.2.58	DellSettings	25
3.2.59	DellSettingsFirmware	25
3.2.60	DellSettingsNicDevice	25
3.2.61	DellStorageInfo	25
3.2.62	DellVirtualDiskInfo	25
3.2.63	Device	25
3.2.64	DevStatus	25
3.2.65	DiskAssertion	25
3.2.66	DiskDevice	25
3.2.67	DiskInfo	25
3.2.68	DiskPartition	25
3.2.69	DiskRaid	25
3.2.70	DiskSetup	25
3.2.71	DiskVolume	25
3.2.72	DiskVolumeGroup	25

---

3.2.73	DrainResult	25
3.2.74	EBSattachAction	25
3.2.75	EBSdetachAction	25
3.2.76	EC2AMI	25
3.2.77	EC2AvailabilityZone	25
3.2.78	EC2EBSStorage	25
3.2.79	EC2EphemeralStorage	25
3.2.80	EC2PrivateCloud	25
3.2.81	EC2Provider	25
3.2.82	EC2Region	25
3.2.83	EC2RegionAMI	25
3.2.84	EC2Settings	25
3.2.85	EC2StaticIP	25
3.2.86	EC2Storage	25
3.2.87	EC2Type	25
3.2.88	EC2VirtualNetworkInterface	25
3.2.89	EntityManagersMD5	25
3.2.90	EthernetSwitch	25
3.2.91	FailoverRole	25
3.2.92	FakeData	25
3.2.93	FSExport	25
3.2.94	FSMount	25
3.2.95	FSPart	25
3.2.96	FSPartAssociation	26
3.2.97	FSPartBasicAssociation	26
3.2.98	FSPartProviderAssociation	26
3.2.99	GenericDevice	26
3.2.100	GenericResource	26
3.2.101	GPUInfo	26
3.2.102	GPUSettings	26
3.2.103	GpuUnit	26
3.2.104	GPUUnitInfo	26
3.2.105	GridEngineJob	26
3.2.106	GridEngineJobQueue	26
3.2.107	GridEngineJobQueueStat	26
3.2.108	GridEngineParallelEnvironment	26
3.2.109	Group	26
3.2.110	GuiCephOsdPoolInfo	26
3.2.111	GuiCephOverview	26
3.2.112	GuiCephPgsInfo	26
3.2.113	GuiClusterOverview	26
3.2.114	GuiCompleteOpenStackOverview	26
3.2.115	GuiDiskUsage	26
3.2.116	GuiGpuUnitOverview	26
3.2.117	GuiHadoopHDFSDetailHBase	26
3.2.118	GuiHadoopHDFSDetailHDFS	26

3.2.119 GuiHadoopHDFSDetailMapreduce . . . . .	26
3.2.120 GuiHadoopHDFSDetailSpark . . . . .	26
3.2.121 GuiHadoopHDFSDetailYarn . . . . .	26
3.2.122 GuiHadoopHDFSDetailZooKeeper . . . . .	26
3.2.123 GuiHadoopHDFSOverview . . . . .	26
3.2.124 GuiJob . . . . .	26
3.2.125 GuiNetSwitchStatus . . . . .	26
3.2.126 GuiNetworkInterface . . . . .	26
3.2.127 GuiNodeOverview . . . . .	26
3.2.128 GuiNodeStatus . . . . .	26
3.2.129 GuiOpenStackOverview . . . . .	26
3.2.130 GuiOpenStackProjectOverview . . . . .	26
3.2.131 GuiOpenStackTenantOverview . . . . .	26
3.2.132 GuiPDUBank . . . . .	26
3.2.133 GuiPDUOutlet . . . . .	26
3.2.134 GuiWorkload . . . . .	26
3.2.135 HadoopBaseConfiguration . . . . .	26
3.2.136 HadoopDataNodeHDFSConfiguration . . . . .	26
3.2.137 HadoopDataNodeRole . . . . .	26
3.2.138 HadoopHBaseClientHDFSConfiguration . . . . .	26
3.2.139 HadoopHBaseClientRole . . . . .	26
3.2.140 HadoopHBaseServerHDFSConfiguration . . . . .	26
3.2.141 HadoopHBaseServerRole . . . . .	26
3.2.142 HadoopHDFS . . . . .	26
3.2.143 HadoopHiveHDFSConfiguration . . . . .	27
3.2.144 HadoopHiveRole . . . . .	27
3.2.145 HadoopJob . . . . .	27
3.2.146 HadoopJobQueue . . . . .	27
3.2.147 HadoopJobQueueStat . . . . .	27
3.2.148 HadoopJobTrackerHDFSConfiguration . . . . .	27
3.2.149 HadoopJobTrackerRole . . . . .	27
3.2.150 HadoopJournalHDFSConfiguration . . . . .	27
3.2.151 HadoopJournalRole . . . . .	27
3.2.152 HadoopKMServerHDFSConfiguration . . . . .	27
3.2.153 HadoopKMServerRole . . . . .	27
3.2.154 HadoopNameNodeHDFSConfiguration . . . . .	27
3.2.155 HadoopNameNodeRole . . . . .	27
3.2.156 HadoopNFSGatewayHDFSConfiguration . . . . .	27
3.2.157 HadoopNFSGatewayRole . . . . .	27
3.2.158 HadoopSecondaryNameNodeHDFSConfiguration . . . . .	27
3.2.159 HadoopSecondaryNameNodeRole . . . . .	27
3.2.160 HadoopSparkMasterHDFSConfiguration . . . . .	27
3.2.161 HadoopSparkMasterRole . . . . .	27
3.2.162 HadoopSparkWorkerHDFSConfiguration . . . . .	27
3.2.163 HadoopSparkWorkerRole . . . . .	27
3.2.164 HadoopSparkYARNHDFSConfiguration . . . . .	27

3.2.165 HadoopSparkYARNRole . . . . .	27
3.2.166 HadoopSqoopHDFSConfiguration . . . . .	27
3.2.167 HadoopSqoopRole . . . . .	27
3.2.168 HadoopTaskTrackerHDFSConfiguration . . . . .	27
3.2.169 HadoopTaskTrackerRole . . . . .	27
3.2.170 HadoopYARNClientHDFSConfiguration . . . . .	27
3.2.171 HadoopYARNClientRole . . . . .	27
3.2.172 HadoopYARNServerHDFSConfiguration . . . . .	27
3.2.173 HadoopYARNServerRole . . . . .	27
3.2.174 HadoopZooKeeperHDFSConfiguration . . . . .	27
3.2.175 HadoopZooKeeperRole . . . . .	27
3.2.176 HAProxyBackendInformation . . . . .	27
3.2.177 HAProxyEntry . . . . .	27
3.2.178 HAProxyEntryBind . . . . .	27
3.2.179 HAProxyFrontendInformation . . . . .	27
3.2.180 HAProxyNodeInformation . . . . .	27
3.2.181 HAProxyRole . . . . .	27
3.2.182 HAProxyServer . . . . .	27
3.2.183 HAProxySharedSettings . . . . .	27
3.2.184 HealthCheck . . . . .	27
3.2.185 HeatMapData . . . . .	27
3.2.186 IBSSwitch . . . . .	27
3.2.187 IniConfigFileCustomizationEntry . . . . .	27
3.2.188 IPCPerm . . . . .	27
3.2.189 IPResource . . . . .	27
3.2.190 Job . . . . .	28
3.2.191 JobQueue . . . . .	28
3.2.192 JobQueuePlaceholder . . . . .	28
3.2.193 JobQueueStat . . . . .	28
3.2.194 KernelModule . . . . .	28
3.2.195 KeyValuePair . . . . .	28
3.2.196 LicenseInfo . . . . .	28
3.2.197 LicenseManagerRole . . . . .	28
3.2.198 LoginRole . . . . .	28
3.2.199 LSFBaseJob . . . . .	28
3.2.200 LSFBaseJobQueue . . . . .	28
3.2.201 LSFBaseJobQueueStat . . . . .	28
3.2.202 LSFClientRole . . . . .	28
3.2.203 LSFJob . . . . .	28
3.2.204 LSFJobQueue . . . . .	28
3.2.205 LSFJobQueueStat . . . . .	28
3.2.206 LSFServerRole . . . . .	28
3.2.207 LustreAlert . . . . .	28
3.2.208 LustreClientMount . . . . .	28
3.2.209 LustreFileSystem . . . . .	28
3.2.210 LustreFileSystemTarget . . . . .	28



---

3.2.211 LustreLog . . . . .	28
3.2.212 LustreOverview . . . . .	28
3.2.213 LustreServer . . . . .	28
3.2.214 LustreServerProfile . . . . .	28
3.2.215 LustreSettings . . . . .	28
3.2.216 LustreTargetMap . . . . .	28
3.2.217 LustreUser . . . . .	28
3.2.218 LustreVolume . . . . .	28
3.2.219 LustreVolumeNode . . . . .	28
3.2.220 MasterNode . . . . .	28
3.2.221 MasterRole . . . . .	28
3.2.222 MemcachedRole . . . . .	28
3.2.223 MemoryInfo . . . . .	28
3.2.224 Metric . . . . .	28
3.2.225 MetricPrmId . . . . .	28
3.2.226 MICHostRole . . . . .	28
3.2.227 MICInfo . . . . .	28
3.2.228 MICNode . . . . .	28
3.2.229 MICNodeCategory . . . . .	28
3.2.230 MICOOverlay . . . . .	28
3.2.231 MICSettings . . . . .	28
3.2.232 MonConf . . . . .	28
3.2.233 MonGlobalConf . . . . .	28
3.2.234 MonHealthConf . . . . .	28
3.2.235 MonitoringRole . . . . .	28
3.2.236 MonMetricConf . . . . .	28
3.2.237 MsgQueue . . . . .	29
3.2.238 MyrinetSwitch . . . . .	29
3.2.239 Network . . . . .	29
3.2.240 NetworkAliasInterface . . . . .	29
3.2.241 NetworkBmcInterface . . . . .	29
3.2.242 NetworkBondInterface . . . . .	29
3.2.243 NetworkBridgeInterface . . . . .	29
3.2.244 NetworkInterface . . . . .	29
3.2.245 NetworkNetMapInterface . . . . .	29
3.2.246 NetworkPhysicalInterface . . . . .	29
3.2.247 NetworkTunnelInterface . . . . .	29
3.2.248 NetworkVLANInterface . . . . .	29
3.2.249 NewNode . . . . .	29
3.2.250 NFSexportAction . . . . .	29
3.2.251 NFSmountAction . . . . .	29
3.2.252 NFSunexportAction . . . . .	29
3.2.253 NFSunmountAction . . . . .	29
3.2.254 Node . . . . .	29
3.2.255 NodeCategory . . . . .	29
3.2.256 NodeGroup . . . . .	29

---

3.2.257 OpenLavaClientRole . . . . .	29
3.2.258 OpenLavaJob . . . . .	29
3.2.259 OpenLavaJobQueue . . . . .	29
3.2.260 OpenLavaJobQueueStat . . . . .	29
3.2.261 OpenLavaServerRole . . . . .	29
3.2.262 OpenStack . . . . .	29
3.2.263 OpenStackApiAgent . . . . .	29
3.2.264 OpenStackApiDomain . . . . .	29
3.2.265 OpenStackApiEndpoint . . . . .	29
3.2.266 OpenStackApiEntity . . . . .	29
3.2.267 OpenStackApiFlavor . . . . .	29
3.2.268 OpenStackApiFloatingIP . . . . .	29
3.2.269 OpenStackApiGroup . . . . .	29
3.2.270 OpenStackApiHostAggregate . . . . .	29
3.2.271 OpenStackApiHypervisor . . . . .	29
3.2.272 OpenStackApiImage . . . . .	29
3.2.273 OpenStackApiNetwork . . . . .	29
3.2.274 OpenStackApiPort . . . . .	29
3.2.275 OpenStackApiProject . . . . .	29
3.2.276 OpenStackApiRole . . . . .	29
3.2.277 OpenStackApiRoleAssignment . . . . .	29
3.2.278 OpenStackApiRouter . . . . .	29
3.2.279 OpenStackApiSecurityGroup . . . . .	29
3.2.280 OpenStackApiServer . . . . .	29
3.2.281 OpenStackApiService . . . . .	29
3.2.282 OpenStackApiSubnet . . . . .	29
3.2.283 OpenStackApiUser . . . . .	29
3.2.284 OpenStackApiVolume . . . . .	30
3.2.285 OpenStackApiVolumeSnapshot . . . . .	30
3.2.286 OpenStackApiVolumeType . . . . .	30
3.2.287 OpenStackBareMetalApiRole . . . . .	30
3.2.288 OpenStackBareMetalConductorRole . . . . .	30
3.2.289 OpenStackBareMetalDiscoverdDNSMasqRole . . . . .	30
3.2.290 OpenStackBareMetalDiscoverdRole . . . . .	30
3.2.291 OpenStackBlockStorage . . . . .	30
3.2.292 OpenStackComputeApiEC2Role . . . . .	30
3.2.293 OpenStackComputeApiMetadataRole . . . . .	30
3.2.294 OpenStackComputeApiRole . . . . .	30
3.2.295 OpenStackComputeConductorRole . . . . .	30
3.2.296 OpenStackComputeRole . . . . .	30
3.2.297 OpenStackComputeSchedulerRole . . . . .	30
3.2.298 OpenStackComputeVNCProxyRole . . . . .	30
3.2.299 OpenStackConfigFileCustomization . . . . .	30
3.2.300 OpenStackDashboardRole . . . . .	30
3.2.301 OpenStackDataProcessingApiRole . . . . .	30
3.2.302 OpenStackDBaaSRole . . . . .	30

---

3.2.303 OpenStackDefaultUserRole . . . . .	30
3.2.304 OpenStackIdentityApiRole . . . . .	30
3.2.305 OpenStackImageApiRole . . . . .	30
3.2.306 OpenStackImageBackend . . . . .	30
3.2.307 OpenStackImageBackendCeph . . . . .	30
3.2.308 OpenStackImageBackendFS . . . . .	30
3.2.309 OpenStackImageRegistryRole . . . . .	30
3.2.310 OpenStackMessageQueueServerRole . . . . .	30
3.2.311 OpenStackNetworkApiRole . . . . .	30
3.2.312 OpenStackNetworkRole . . . . .	30
3.2.313 OpenStackNodeRole . . . . .	30
3.2.314 OpenStackNovaImageBackend . . . . .	30
3.2.315 OpenStackNovaImageBackendCeph . . . . .	30
3.2.316 OpenStackNovaImageBackendCow . . . . .	30
3.2.317 OpenStackObjectAccountRole . . . . .	30
3.2.318 OpenStackObjectApiRole . . . . .	30
3.2.319 OpenStackObjectContainerRole . . . . .	30
3.2.320 OpenStackObjectStoreRole . . . . .	30
3.2.321 OpenStackOrchestrationApiRole . . . . .	30
3.2.322 OpenStackOrchestrationRole . . . . .	30
3.2.323 OpenStackSettings . . . . .	30
3.2.324 OpenStackSettingsAdvanced . . . . .	30
3.2.325 OpenStackSettingsCMDaemonInteractions . . . . .	30
3.2.326 OpenStackSettingsCompute . . . . .	30
3.2.327 OpenStackSettingsCredentials . . . . .	30
3.2.328 OpenStackSettingsDatabase . . . . .	30
3.2.329 OpenStackSettingsLogging . . . . .	30
3.2.330 OpenStackSettingsNetworking . . . . .	30
3.2.331 OpenStackSettingsPorts . . . . .	31
3.2.332 OpenStackSettingsQuota . . . . .	31
3.2.333 OpenStackSettingsUserPortal . . . . .	31
3.2.334 OpenStackSettingsUsers . . . . .	31
3.2.335 OpenStackStorage . . . . .	31
3.2.336 OpenStackTelemetryAgentCentralRole . . . . .	31
3.2.337 OpenStackTelemetryAgentComputeRole . . . . .	31
3.2.338 OpenStackTelemetryAgentIpmiRole . . . . .	31
3.2.339 OpenStackTelemetryAgentNotificationRole . . . . .	31
3.2.340 OpenStackTelemetryAlarmEvaluatorRole . . . . .	31
3.2.341 OpenStackTelemetryAlarmNotifierRole . . . . .	31
3.2.342 OpenStackTelemetryApiRole . . . . .	31
3.2.343 OpenStackTelemetryCollectorRole . . . . .	31
3.2.344 OpenStackUserRole . . . . .	31
3.2.345 OpenStackUserSettings . . . . .	31
3.2.346 OpenStackVolumeApiRole . . . . .	31
3.2.347 OpenStackVolumeBackend . . . . .	31
3.2.348 OpenStackVolumeBackendCeph . . . . .	31

3.2.349 OpenStackVolumeBackendNFS . . . . .	31
3.2.350 OpenStackVolumeBackupBackend . . . . .	31
3.2.351 OpenStackVolumeBackupBackendCeph . . . . .	31
3.2.352 OpenStackVolumeBackupRole . . . . .	31
3.2.353 OpenStackVolumeRole . . . . .	31
3.2.354 OpenStackVolumeSchedulerRole . . . . .	31
3.2.355 OsapiPortIP . . . . .	31
3.2.356 OsapiSecurityGroupRule . . . . .	31
3.2.357 OsapiSubnetAllocationPool . . . . .	31
3.2.358 OSService . . . . .	31
3.2.359 OSServiceArray . . . . .	31
3.2.360 OSServiceConfig . . . . .	31
3.2.361 Partition . . . . .	31
3.2.362 PBSJob . . . . .	31
3.2.363 PBSJobQueue . . . . .	31
3.2.364 PBSJobQueueStat . . . . .	31
3.2.365 PbsProClientRole . . . . .	31
3.2.366 PbsProJob . . . . .	31
3.2.367 PbsProJobQueue . . . . .	31
3.2.368 PbsProJobQueueStat . . . . .	31
3.2.369 PbsProServerRole . . . . .	31
3.2.370 PDUPort . . . . .	31
3.2.371 PhysicalNode . . . . .	31
3.2.372 PowerDistributionUnit . . . . .	31
3.2.373 PowerStatus . . . . .	31
3.2.374 Process . . . . .	31
3.2.375 Processor . . . . .	31
3.2.376 Profile . . . . .	31
3.2.377 ProgramRunnerInput . . . . .	31
3.2.378 ProgramRunnerKill . . . . .	32
3.2.379 ProgramRunnerOutput . . . . .	32
3.2.380 ProgramRunnerStatus . . . . .	32
3.2.381 ProvisioningNodeStatus . . . . .	32
3.2.382 ProvisioningProcessorJob . . . . .	32
3.2.383 ProvisioningRequestStatus . . . . .	32
3.2.384 ProvisioningRole . . . . .	32
3.2.385 ProvisioningStatus . . . . .	32
3.2.386 Puppet . . . . .	32
3.2.387 PuppetApplyResult . . . . .	32
3.2.388 PuppetApplySession . . . . .	32
3.2.389 PuppetClass . . . . .	32
3.2.390 PuppetRole . . . . .	32
3.2.391 PuppetRunInfo . . . . .	32
3.2.392 Rack . . . . .	32
3.2.393 RackSensor . . . . .	32
3.2.394 RadosGatewayRole . . . . .	32

---

3.2.395 RateElem . . . . .	32
3.2.396 ReadMonDataId . . . . .	32
3.2.397 ReadMonDataOutput . . . . .	32
3.2.398 RemoteMonConf . . . . .	32
3.2.399 RemoteMonMetricConf . . . . .	32
3.2.400 RemoteNodeInstallerInteraction . . . . .	32
3.2.401 RemoteSetupExecution . . . . .	32
3.2.402 RemoteThreshold . . . . .	32
3.2.403 ResourcePool . . . . .	32
3.2.404 ResourcePoolStatus . . . . .	32
3.2.405 Role . . . . .	32
3.2.406 RunJobAction . . . . .	32
3.2.407 S3DataDownload . . . . .	32
3.2.408 S3DataUpload . . . . .	32
3.2.409 S3ResultsDownload . . . . .	32
3.2.410 S3ResultsUpload . . . . .	32
3.2.411 S3Transfer . . . . .	32
3.2.412 Semaphore . . . . .	32
3.2.413 Sensor . . . . .	32
3.2.414 Session . . . . .	32
3.2.415 SGEClientRole . . . . .	32
3.2.416 SGEJob . . . . .	32
3.2.417 SGEJobQueue . . . . .	32
3.2.418 SGEJobQueueStat . . . . .	32
3.2.419 SGEParallelEnvironment . . . . .	32
3.2.420 SGEServerRole . . . . .	32
3.2.421 SharedMemory . . . . .	32
3.2.422 SlaveMonotonicElem . . . . .	32
3.2.423 SlaveMonSnapshot . . . . .	32
3.2.424 SlaveNode . . . . .	32
3.2.425 SlaveRateElem . . . . .	33
3.2.426 SlurmClientRole . . . . .	33
3.2.427 SlurmJob . . . . .	33
3.2.428 SlurmJobQueue . . . . .	33
3.2.429 SlurmJobQueueStat . . . . .	33
3.2.430 SlurmServerRole . . . . .	33
3.2.431 SoftwareImage . . . . .	33
3.2.432 SoftwareImageProxy . . . . .	33
3.2.433 StartStorageNodeAction . . . . .	33
3.2.434 StateElem . . . . .	33
3.2.435 StaticRoute . . . . .	33
3.2.436 StatisticData . . . . .	33
3.2.437 StopStorageNodeAction . . . . .	33
3.2.438 StorageNodePolicy . . . . .	33
3.2.439 StorageRole . . . . .	33
3.2.440 StringListObject . . . . .	33

3.2.441 SubnetManagerRole . . . . .	33
3.2.442 Switch . . . . .	33
3.2.443 SwitchPort . . . . .	33
3.2.444 SysInfoCollector . . . . .	33
3.2.445 SysMonotonicWithId . . . . .	33
3.2.446 SysRateWithId . . . . .	33
3.2.447 ThreshAction . . . . .	33
3.2.448 ThreshActionConf . . . . .	33
3.2.449 Threshold . . . . .	33
3.2.450 Ticket . . . . .	33
3.2.451 TorqueClientRole . . . . .	33
3.2.452 TorqueJob . . . . .	33
3.2.453 TorqueJobQueue . . . . .	33
3.2.454 TorqueJobQueueStat . . . . .	33
3.2.455 TorqueServerRole . . . . .	33
3.2.456 UCSAdaptorEthCompQueueProfile . . . . .	33
3.2.457 UCSAdaptorEthGenProfile . . . . .	33
3.2.458 UCSAdaptorEthInterruptProfile . . . . .	33
3.2.459 UCSAdaptorEthOffloadProfile . . . . .	33
3.2.460 UCSAdaptorEthRecvQueueProfile . . . . .	33
3.2.461 UCSAdaptorEthUSNICProfile . . . . .	33
3.2.462 UCSAdaptorEthWorkQueueProfile . . . . .	33
3.2.463 UCSAdaptorExtEthIf . . . . .	33
3.2.464 UCSAdaptorExtIpV6RssHashProfile . . . . .	33
3.2.465 UCSAdaptorFcCdbWorkQueueProfile . . . . .	33
3.2.466 UCSAdaptorFcErrorRecoveryProfile . . . . .	33
3.2.467 UCSAdaptorFcGenProfile . . . . .	33
3.2.468 UCSAdaptorFcInterruptProfile . . . . .	33
3.2.469 UCSAdaptorFcPortFLogiProfile . . . . .	33
3.2.470 UCSAdaptorFcPortPLogiProfile . . . . .	33
3.2.471 UCSAdaptorFcPortProfile . . . . .	33
3.2.472 UCSAdaptorFcRecvQueueProfile . . . . .	34
3.2.473 UCSAdaptorFcWorkQueueProfile . . . . .	34
3.2.474 UCSAdaptorHostEthIf . . . . .	34
3.2.475 UCSAdaptorHostFcIf . . . . .	34
3.2.476 UCSAdaptorIpV4RssHashProfile . . . . .	34
3.2.477 UCSAdaptorIpV6RssHashProfile . . . . .	34
3.2.478 UCSAdaptorPortProfiles . . . . .	34
3.2.479 UCSAdaptorRssProfile . . . . .	34
3.2.480 UCSBase . . . . .	34
3.2.481 UCSBiosBootDev . . . . .	34
3.2.482 UCSBiosBootDevGrp . . . . .	34
3.2.483 UCSBiosSettings . . . . .	34
3.2.484 UCSBiosVfAdjacentCacheLinePrefetch . . . . .	34
3.2.485 UCSBiosVfAltitude . . . . .	34
3.2.486 UCSBiosVfASPMsupport . . . . .	34

3.2.487 UCSBiosVfConsoleRedirection . . . . .	34
3.2.488 UCSBiosVfCoreMultiProcessing . . . . .	34
3.2.489 UCSBiosVfCPUEnergyPerformance . . . . .	34
3.2.490 UCSBiosVfCPUFrequencyFloor . . . . .	34
3.2.491 UCSBiosVfCPUPerformance . . . . .	34
3.2.492 UCSBiosVfCPUPowerManagement . . . . .	34
3.2.493 UCSBiosVfDCUPrefetch . . . . .	34
3.2.494 UCSBiosVfDemandScrub . . . . .	34
3.2.495 UCSBiosVfDirectCacheAccess . . . . .	34
3.2.496 UCSBiosVfDRAMClockThrottling . . . . .	34
3.2.497 UCSBiosVfDramRefreshRate . . . . .	34
3.2.498 UCSBiosVfEnhancedIntelSpeedStepTech . . . . .	34
3.2.499 UCSBiosVfExecuteDisableBit . . . . .	34
3.2.500 UCSBiosVfFRB2Enable . . . . .	34
3.2.501 UCSBiosVfHardwarePrefetch . . . . .	34
3.2.502 UCSBiosVfIntelHyperThreadingTech . . . . .	34
3.2.503 UCSBiosVfIntelTurboBoostTech . . . . .	34
3.2.504 UCSBiosVfIntelVirtualizationTechnology . . . . .	34
3.2.505 UCSBiosVfIntelVTForDirectedIO . . . . .	34
3.2.506 UCSBiosVfLegacyUSBsupport . . . . .	34
3.2.507 UCSBiosVfLOMPortOptionROM . . . . .	34
3.2.508 UCSBiosVfLvDIMMSupport . . . . .	34
3.2.509 UCSBiosVfMemoryInterleave . . . . .	34
3.2.510 UCSBiosVfMemoryMappedIOAbove4GB . . . . .	34
3.2.511 UCSBiosVfNUMAOptimized . . . . .	34
3.2.512 UCSBiosVfOnboardStorage . . . . .	34
3.2.513 UCSBiosVfOnboardStorageSWStack . . . . .	34
3.2.514 UCSBiosVfOSBootWatchdogTimer . . . . .	34
3.2.515 UCSBiosVfOSBootWatchdogTimerPolicy . . . . .	34
3.2.516 UCSBiosVfOSBootWatchdogTimerTimeout . . . . .	34
3.2.517 UCSBiosVfPatrolScrub . . . . .	34
3.2.518 UCSBiosVfPCIOptionROMs . . . . .	34
3.2.519 UCSBiosVfPCISlotOptionROMEnable . . . . .	35
3.2.520 UCSBiosVfProcessorC1E . . . . .	35
3.2.521 UCSBiosVfProcessorC6Report . . . . .	35
3.2.522 UCSBiosVfPStateCoordType . . . . .	35
3.2.523 UCSBiosVfQPIConfig . . . . .	35
3.2.524 UCSBiosVfSelectMemoryRASConfiguration . . . . .	35
3.2.525 UCSBiosVfTPMSupport . . . . .	35
3.2.526 UCSBiosVfUCSMBootOrderRuleControl . . . . .	35
3.2.527 UCSBiosVfUSBEmulation . . . . .	35
3.2.528 UCSBiosVfUSBPortsConfig . . . . .	35
3.2.529 UCSBiosVfVgaPriority . . . . .	35
3.2.530 UCSCommNtpProvider . . . . .	35
3.2.531 UCSCommSyslog . . . . .	35
3.2.532 UCSCommSyslogClient . . . . .	35

---

3.2.533 UCSEquipmentIndicatorLed . . . . .	35
3.2.534 UCSEquipmentLocatorLed . . . . .	35
3.2.535 UCSFaultInst . . . . .	35
3.2.536 UCSFirmwareRunning . . . . .	35
3.2.537 UCSInfo . . . . .	35
3.2.538 UCSLogs . . . . .	35
3.2.539 UCSLsbootDef . . . . .	35
3.2.540 UCSLsbootEfi . . . . .	35
3.2.541 UCSLsbootLan . . . . .	35
3.2.542 UCSLsbootStorage . . . . .	35
3.2.543 UCSLsbootVirtualMedia . . . . .	35
3.2.544 UCSStatus . . . . .	35
3.2.545 UGECgroupsSettings . . . . .	35
3.2.546 UGEClientRole . . . . .	35
3.2.547 UGEJob . . . . .	35
3.2.548 UGEJobQueue . . . . .	35
3.2.549 UGEJobQueueStat . . . . .	35
3.2.550 UGEParallelEnvironment . . . . .	35
3.2.551 UGEServerRole . . . . .	35
3.2.552 User . . . . .	35
3.2.553 Validation . . . . .	35
3.2.554 VersionInfo . . . . .	35
3.2.555 VirtualNode . . . . .	35
3.2.556 VirtualNodeSettings . . . . .	35
3.2.557 VirtualSMPNode . . . . .	35
3.2.558 VScaleMPSettings . . . . .	35
3.2.559 VsmSettings . . . . .	35
3.2.560 WillChange . . . . .	35
3.2.561 XeonPhiSettings . . . . .	35
3.3 JSON Examples . . . . .	35



# Preface

Welcome to the *Developer Manual* for Bright Cluster Manager 7.3.

## 0.1 About This Manual

This manual is aimed at helping developers who would like to program the Bright Cluster Manager in order to enhance or alter its functionality. It is not intended for end users who simply wish to submit jobs that run programs to workload managers, which is discussed in the *User Manual*. The developer is expected to be reasonably familiar with the parts of the *Administrator Manual* that is to be dealt with—primarily CMDaemon, of which `cmsh` and `cmgui` are the front ends.

This manual discusses the Python API to CMDaemon, and also covers how to program for metric collections.

## 0.2 About The Manuals In General

Regularly updated versions of the Bright Cluster Manager 7.3 manuals are available on updated clusters by default at `/cm/shared/docs/cm`. The latest updates are always online at <http://support.brightcomputing.com/manuals>.

- The *Administrator Manual* describes the general management of the cluster.
- The *Installation Manual* describes installation procedures for a basic cluster.
- The *User Manual* describes the user environment and how to submit jobs for the end user.
- The *Cloudbursting Manual* describes how to deploy the cloud capabilities of the cluster.
- The *Developer Manual* has useful information for developers who would like to program with Bright Cluster Manager.
- The *OpenStack Deployment Manual* describes how to deploy OpenStack with Bright Cluster Manager.
- The *Big Data Deployment Manual* describes how to deploy Big Data with Bright Cluster Manager.
- The *UCS Deployment Manual* describes how to deploy the Cisco UCS server with Bright Cluster Manager.
- The *Machine Learning Manual* describes how to install and configure machine learning capabilities with Bright Cluster Manager.

If the manuals are downloaded and kept in one local directory, then in most pdf viewers, clicking on a cross-reference in one manual that refers to a section in another manual opens and displays that section in the second manual. Navigating back and forth between documents is usually possible with keystrokes or mouse clicks.

For example: `<Alt>-<Backarrow>` in Acrobat Reader, or clicking on the bottom leftmost navigation button of `xpdf`, both navigate back to the previous document.

The manuals constantly evolve to keep up with the development of the Bright Cluster Manager environment and the addition of new hardware and/or applications. The manuals also regularly incorporate customer feedback. Administrator and user input is greatly valued at Bright Computing. So any comments, suggestions or corrections will be very gratefully accepted at [manuals@brightcomputing.com](mailto:manuals@brightcomputing.com).

### 0.3 Getting Administrator-Level Support

If the reseller from whom Bright Cluster Manager was bought offers direct support, then the reseller should be contacted.

Otherwise the primary means of support is via the website <https://support.brightcomputing.com>. This allows the administrator to submit a support request via a web form, and opens up a trouble ticket. It is a good idea to try to use a clear subject header, since that is used as part of a reference tag as the ticket progresses. Also helpful is a good description of the issue. The followup communication for this ticket goes via standard e-mail. Section 11.2 of the *Administrator Manual* has more details on working with support.

### 0.4 Getting Developer-Level Support

Developer support is given free, within reason. For more extensive support, Bright Computing can be contacted in order to arrange a support contract.

### 0.5 Getting Professional Services

Bright Computing normally differentiates between professional services (customer asks Bright Computing to do something or asks Bright Computing to provide some service) and support (customer has a question or problem that requires an answer or resolution). Professional services can be provided after consulting with the reseller, or the Bright account manager.

# 1

## Bright Cluster Manager Python API

This chapter introduces the Python API of Bright Cluster Manager. For a head node `bright72`, the API reference documentation for all available objects is available in a default cluster via browser access to the URL:

```
https://bright72/userportal/downloads/python
```

The preceding access is via the User Portal (section 10.9 of the *Administrator Manual*).

The documentation is also available directly on the head node itself at:

```
file:///cm/local/docs/cmd/python/index.html
```

### 1.1 Installation

The Python cluster manager bindings are pre-installed on the head node.

#### 1.1.1 Windows Clients

For windows clients, Python version 2.5.X is needed. Newer versions of Python do not work with the API.

For Windows a redistributable package is supplied in the `pythoncm-dist` RPM package installed on the cluster. The file at

```
/cm/shared/apps/pythoncm/dist/windows-pythoncm.7.3.r15673.zip
```

—the exact version number may differ—is copied to the Windows PC and unzipped.

A Windows shell (`cmd.exe`) is opened to the directory where the Python bindings are. The `headnodeinfo.py` example supplied with the unzipped files has a line that has the following format:

```
cluster = clustermanager.addCluster(<parameters>);
```

where `<parameters>` is either:

```
'<URL>', '<PEMauth1>', '<PEMauth2>'
```

or

```
'<URL>', '<PFXauth>', '"', '<password>'
```

The `<parameters>` entry is edited as follows:

- the correct hostname is set for the `<URL>` entry. By default it is set to `https://localhost:8081`
- If PEM key files are to be used for client authentication,

- `<PEMauth1>` is set to path of `cert.pem`
- `<PEMauth2>` is set to the path of `cert.key`
- If a PFX file is used for client authentication,
  - `<PFXauth>` is set to path of `admin.pfx`
  - `<password>` is set to the password

To verify everything is working, it can be run as follows:

```
c:\python25\python headnodeinfo.py
```

### 1.1.2 Linux Clients

For Linux clients, a redistributable source package is supplied in the `pythoncm-dist` package installed on the cluster. The file at `/cm/shared/apps/pythoncm/dist/pythoncm-7.3-r18836-src.tar.bz2`—the exact version number may differ—is copied and untarred to any directory.

The `build.sh` script is then run to compile the source. About 4GB of memory is usually needed for compilation, and additional packages may be required for compilation to succeed. A list of packages needed to build Python cluster manager bindings can be found in the `README` file included with the package.

The `headnodeinfo.py` example supplied with the untarred files is edited as for in the earlier windows client example, for the `clustermanager.addCluster` line.

The path to the remote cluster manager library is added:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:remotecm
```

To verify everything is working, the following can be run:

```
python ./headnodeinfo.py
```

## 1.2 Examples

A set of examples can be found in `/cm/local/examples/cmd/python/` on the head node of the cluster.

### 1.2.1 First Program

A Python script is told to use the cluster manager bindings by importing `pythoncm` at the start of the script:

```
import pythoncm
```

If not working on the cluster, the administrator needs to set the path where the shared libraries can be found (`pythoncm.so` in Linux, or `python.pyd` in windows). This is done by adding the following to the start of the script:

```
import sys
sys.path.append(".") # path to pythoncm.so/python.pyd
```

Python cluster manager bindings allow for simultaneous connections to several clusters. For this reason the first thing to do is to create a `ClusterManager` object:

```
clustermanager = pythoncm.ClusterManager()
```

A connection to a cluster can now be made. There are two possible ways of connecting.

The first is using the certificate and private key file that `cmsh` uses by default when it authenticates from the head node.

```
cluster = clustermanager.addCluster('https://mycluster:8081', \
'/root/.cm/admin.pem', '/root/.cm/admin.key');
```

The second way uses the password protected `admin.pfx` file, which is generated with the `cmd -c` command. A Python script could ask for the password and store it in a variable for increased security.

```
cluster = clustermanager.addCluster('https://mycluster:8081', \
'/root/.cm/cmgui/admin.pfx', '', '<password>');
```

Having defined the cluster, a connection can now be made to it:

```
isconnected = cluster.connect()
if !isconnected:
    print "Unable to connect"
    print cluster.getLastError()
    exit(1)
```

If a connection cannot be made, the function `cluster.connect()` returns false. The function `cluster.getLastError()` shows details about the problem. The two most likely problems are due to a wrong password setting or a firewall settings issue.

Similar to `cmgui` and `cmsh`, the cluster object contains a local cache of all objects. This cache will be filled automatically when the connection is established. All changes to properties will be done on these local copies and will be lost after the Python scripts exits, unless a `commit` operation is done.

The most common operation is finding specific objects in the cluster.

```
active = cluster.find('active')
if active == None:
    print "Unable to find active head node"
    exit(1)
else:
    print "Hostname of the active head node is %s" % active.hostname
```

If creating an automated script that runs at certain times, then it is highly recommended to check if objects can be found. During a failover, for instance, there will be a period over a few minutes in which the active head node will not be set.

It is good practice to disconnect from the cluster at the end of the script.

```
cluster.disconnect()
```

When connecting to a cluster with a failover setup, it is the shared IP address that should be connected to, and not the fixed IP address of either of the head nodes.

## 1.3 Methods And Properties

### 1.3.1 Viewing All Properties And Methods

All properties visible in `cmsh` and `cmgui` are also accessible from Python cluster manager bindings. The easiest way to get an overview of the methods and properties of an object is to define the following function:

```
import re
def dump(obj):
    print "--- DUMP ---"
    for attr in dir(obj):
        p = re.compile('^__.*__$')
        if not p.match(attr):
            print "%s = %s" % (attr, getattr(obj, attr))
```

An overview of all properties and methods for the active head node can be obtained with:

```
active = cluster.find('active')
dump(active)
```

### 1.3.2 Property Lists

Most properties are straightforward and their names are almost identical to the `cmsh` equivalent.

For instance:

```
node.mac = '00:00:00:00:00:00'
category.softwareimage = cluster.find('testimage')
```

Properties that contain lists, like `node.roles`, `node.interfaces`, `category.fsmounts` and several others, are trickier to deal with. While iterating over a list property is simple enough:

```
for role in node.roles:
    print role.name
```

because of an implementation restriction, adding a new role requires that a local copy of the roles list be made:

```
roles = node.roles
provisioningrole = pythoncm.ProvisioningRole() # Create a new pro\
                                              visioning role object
roles.append(provisioningrole)
node.roles = roles # This will update the internal\
                  roles list with the local copy
```

### 1.3.3 Creating New Objects

Creating a new node can be done with:

```
node = pythoncm.Node()
```

This is valid command, but fairly useless because a node has to be a `MasterNode`, `PhysicalNode` or `VirtualSMPNode`. So to create a normal compute or login node, the object is created as follows:

```
node = pythoncm.PhysicalNode()
```

The first thing to do after creating a new object is to add it to a cluster.

```
cluster.add(node)
```

It is impossible to add one node to more than one cluster.

After the node has been added its properties can be set. In `cmsh` and `cmgui` this is semi-automated, but in Python cluster manager bindings it has to be done by hand.

```
node.hostname = 'node001'
node.partition = cluster.find('base')
node.category = cluster.find('default')
```

Similar to the `node` object, a `NetworkInterface` object has several subtypes: `NetworkPhysicalInterface`, `NetworkVLANInterface`, `NetworkAliasInterface`, `NetworkBondInterface`, and `NetworkIPMIInterface`.

```
interface = pythoncm.NetworkPhysicalInterface()
interface.name = 'eth0'
interface.ip = '10.141.0.1'
interface.network = cluster.find('internalnet')
node.interfaces = [interface]
node.provisioningInterface = interface
```

Having set the properties of the new node, it can now be committed.

```
cr = node.commit()
```

If a commit fails for some reason, the reason can be found:

```
if not cr.result:
    print "Commit of %s failed:" % node.resolveName()
    for j in range(cr.count):
        print cr.getValidation(j).msg
```

### 1.3.4 List Of Objects

In the following lists of objects:

- Objects marked with (\*) cannot be used
- Trees marked with (+) denote inheritance

#### Roles

```
Role (*)
+ BackupRole
+ BootRole
+ DatabaseRole
+ EthernetSwitch
+ LoginRole
+ LSFClientRole
+ LSFServerRole
+ MasterRole
+ PbsProClientRole
+ PbsProServerRole
+ ProvisioningRole
+ SGEClientRole
+ SGEServerRole
+ SlurmClientRole
+ SlurmServerRole
+ SubnetManagerRole
+ TorqueClientRole
+ TorqueServerRole
```

#### Devices

```
Device (*)
+ Chassis
+ GpuUnit
+ GenericDevice
+ PowerDistributionUnit
+ Switch (*)
  + EthernetSwitch
  + IBSwitch
  + MyrinetSwitch
Node (*)
+ FSExport
+ FSMount
+ MasterNode
+ SlaveNode (*)
  + PhysicalNode
  + VirtualSMPNode
```

#### Network Interfaces

```
NetworkInterface (*)
+ NetworkAliasInterface
+ NetworkBondInterface
```

- + NetworkIpmiInterface
- + NetworkPhysicalInterface
- + NetworkVLANInterface

**Information Objects**

- ClusterSetup
- GuiClusterOverview
- GuiCephOverview
- GuiHadoopHDFSOverview
- GuaOpenStackOverview
- GuiOpenStackTenantOverview
- GuiGpuUnitOverview
- GuiNodeOverview
- GuiNodeStatus
- LicenseInfo
- SysInfoCollector
- VersionInfo

**Monitoring Configuration Objects**

- MonConf
- ConsolidatorConf
- MonHealthConf
- HealthCheck
- MonMetricConf
- ThreshActionConf
- ThreshAction
- Threshold

**LDAP Objects**

- User
- Group

**Category Objects**

- Category
- FSExport
- FSMount

**Miscellaneous Objects**

- SoftwareImage
  
- KernelModule
  
- Network
  
- NodeGroup
  
- Partition
- + BurnConfig
- Rack



### 1.3.5 Useful Methods

#### For The Cluster Object:

Name	Description
<code>find(&lt;name&gt;)</code>	Find the object with a given name, <i>&lt;name&gt;</i>
<code>find(&lt;name&gt;, &lt;type&gt;)</code>	Because it is possible to give a category and node the same name, sometimes the type <i>&lt;type&gt;</i> of the object needs to be specified too
<code>getAll(&lt;type&gt;)</code>	Get a list of all objects of a given type: e.g. device, category
<code>activeMaster()</code>	Get the active master object
<code>passiveMaster()</code>	Get the active master object
<code>overview()</code>	Get all the data shown in the <code>cmgui</code> cluster overview
<code>add(&lt;object&gt;)</code>	Add a newly created object <i>&lt;object&gt;</i> to the cluster. Only after an object is added can it be used
<code>pexec(&lt;nodes&gt;, &lt;command&gt;)</code>	Execute a command <i>&lt;command&gt;</i> on one or more nodes

#### For Any Object:

Name	Description
<code>commit()</code>	Save changes to the cluster
<code>refresh()</code>	Undo all changes and restore the object to its last saved state
<code>remove()</code>	Remove an object from the cluster
<code>clone()</code>	Make an identical copy. The newly created object is not added to a cluster yet

#### For Any Device:

Name	Description
<code>close()</code>	Close a device
<code>open()</code>	Open a device
<code>powerOn()</code>	Power on a device
<code>powerOff()</code>	Power off a device
<code>powerReset()</code>	Power reset a device
<code>latestMonitoringData()</code>	Return a list of the most recent monitoring data

#### For Any Node:

Name	Description
<code>overview()</code>	Get the data displayed in the <code>cmgui</code> node overview tab
<code>sysinfo()</code>	Get the data displayed in the <code>cmgui</code> node system information tab
<code>pexec(&lt;command&gt;)</code>	Execute a command

### 1.3.6 Useful Example Program

In the directory `/cm/local/examples/cmd/python` are some example programs using the python API.

One of these is `printall.py`. It displays values for objects in an easily viewed way. With `all` as the argument, it displays resource objects defined in a list in the program. The objects are 'Partition', 'MasterNode', 'SlaveNode', 'Category', 'SoftwareImage', 'Network', 'NodeGroup'. The output is displayed something like (some output elided):

### Example

```
[root@bright72 ~]# cd /cm/local/examples/cmd/python
[root@bright72 python]# ./printall all
Partition base
+- revision .....
| name ..... base
| clusterName ..... Bright 7.3 Cluster
...
| burnConfigs
| +- revision .....
| | name ..... default
| | description ..... Standard burn test.
| | configuration ..... < 2780 bytes >
| +- revision .....
| | name ..... long-hpl
...
| provisioningInterface ..... None
| fsmounts ..... < none >
| fsexports
| +- revision .....
| | name ..... /cm/shared@internalnet
| | path ..... /cm/shared
| | hosts ..... !17179869185!
...
Category default
+- revision .....
| name ..... default
| softwareImage ..... default-image
| defaultGateway ..... 10.141.255.253
| nameServers ..... < none >
...
```

The values of a particular resource-level object, such as `softwareimage`, can be viewed by specifying it as the argument:

### Example

```
[root@bright72 python]# ./printall.py softwareimage
softwareimage default-image
+- revision .....
| name ..... default-image
| path ..... /cm/images/default-image
| originalImage ..... 0
| kernelVersion ..... 2.6.32-431.11.2.el6.x86_64
| kernelParameters ..... rdblacklist=nouveau
| creationTime ..... 1398679806
| modules
| +- revision .....
| | name ..... xen-netfront
...
```

```

| +- revision .....
| | name ..... hpilo
| | parameters .....
| enableSOL ..... False
| SOLPort ..... ttyS1
| SOLSpeed ..... 115200
| SOLFlowControl ..... True
| notes .....
| fspart ..... 98784247812
| bootfspart ..... 98784247813
...
[root@bright72 python]#

```

## 1.4 The Workload Management API

The workload management API allows the submission of jobs, the retrieval of information on jobs and queues, and the management of jobs and queues. The methods described in this section are a part of the `cmjob` service. They can also be accessed via the `Cluster` object, with exception of the `getParentJobs` and `getJobsSlice` methods.

Workload management examples for a particular workload manager `<wlm>` in Python can be found on the head node in the directory:

```
/cm/local/examples/cmd/python/workload-<wlm>.py
```

Here, `<wlm>` can take the values `torque`, `slurm`, `sgc`, `pbspro`, `openlava`, or `lsf`. The examples define a job, with different job properties associated with different workload managers. With the right properties set, the job can be submitted and the submitted job outputs are printed to STDOUT.

Details of entities and their properties can be found in the CMDaemon API reference.

### 1.4.1 Job Submission

Job submission is performed with the `submitJob` method. Its only argument is the `Job` entity that provides the properties and resource requirements of the job that is submitted.

Each workload manager uses its own job properties format, although they usually behave in a similar way. The following table shows the correspondence between `Job` entity parameters and the submission parameters for each workload manager.

Parameter	Slurm	PBS Pro Torque	LSF openlava	UGE OGS (SGE)
queue	-p	-q	-q	-q
jobname	-J	-N	-J	-N

...continues

*...continued*

<b>Parameter</b>	<b>Slurm</b>	<b>PBS Pro Torque</b>	<b>LSF openlava</b>	<b>UGE OGS (SGE)</b>
account	-A	-A	N/A	-A
project	N/A	-P	-P	-P
rundirectory	-D	-w	N/A	-wd
username	Job script is submitted by this user			
groupname	Job script is submitted with group permissions of this user			
priority	--nice	-p	-sp	-p
stdinfile	-i	N/A	-i	-i
stdoutfile	-o	-o	-o	-o
stderrfile	-e	-e	-e	-e
dependencies	-d	-W depend=	-w	--hold_jid
mailNotify	Enables passing other email options, not used directly			
mailOptions	--mail-type	-m	-B	-m
mailList	--mail-user	-M	-u	-M
resourceList	-C	-l	-R	-l

*...continues*

...continued

Parameter	Slurm	PBS Pro Torque	LSF openlava	UGE OGS (SGE)
maxWallClock	-t	-l walltime=	-c	-l h_rt=
numberOfProcesses	-n	mpiprocs= ppn=	-n	-pe
numberOfNodes	-N	-l select=	-R 'span[hosts=]'	N/A
nodes	-w	-l select=	-m	-l hostname=
environmentVariables	All additional environment variables are passed to the job			
commandLineInterpreter	Interpreter path is added as a first line into the jobscript			
executable	Added as a command at the end of a new created jobscript.			
arguments	Appended to <code>executable</code> line			
modules	Module files will be added to job script environment			
userdefined	These lines are added into the jobscript before the <code>executable</code> line			
scriptFile	If scriptfile is specified, then only is it submitted			
debug	Return debug info (without submission), including generated script			

## Notes:

1. In the case of LSF and OpenLava, the `rundirectory` parameter of the `Job` entity is converted into a `cd` command line, that is added to the job script before any commands.
2. The executable file path and its arguments are translated to a single line in the job script. If more complex commands are required then the parameter `userdefined` should be used instead of `executable` and `arguments`. If `userdefined` is not an empty list, then `executable` and `arguments` are ignored.

### 1.4.2 Job Information And Management

For job manipulation the following functions are used. In these functions, the parameter `<scheduler>` is the name of the workload manager that the operation is applied to, and takes a value of `slurm`, `uge`, `sge`, `openlava`, `lsf`, `torque` or `pbspro`. The parameter `<JobID>` is a string in a format related to that particular workload manager.

**getJobs(<scheduler>):** returns `Job` entities for the specified scheduler. This function triggers a call to the workload manager utility. The workload manager utility is, for example, `qstat` in the case of SGE or Torque, and `scontrol` in the case of Slurm. In profiles (section 6.4 of the *Administrator Manual*), `GET_JOB_TOKEN` is needed to be able to get all the jobs, while `GET_OWN_JOB_TOKEN` is needed to get just all the jobs belonging to the user making the call.

**getJob(<scheduler>, <JobID>):** returns a job by job ID. `GET_JOB_TOKEN` is needed to be able to get any job, and `GET_OWN_JOB_TOKEN` is needed to be able to get just the job belonging to the user making the call.

**removeJob(<scheduler>, <JobID>):** removes the job by job ID and returns the result of job removal. `UPDATE_JOB_TOKEN` is needed to be able to remove any job, and `UPDATE_OWN_JOB_TOKEN` is needed to be able to remove just the job belonging to the user making the call.

**getJobsSlice(<scheduler>, <start>, <maxCount>, <parentID>, <allUsers>):** returns jobs at the position `<start>` in the global list (sorted by job ID), but only up to `<maxCount>` items. That is, if the value of the parameter `<start>` is a number `n`, then jobs starting from the `n`th item in the global list are returned, up to `<maxCount>` times. `<parentID>` is a method to group jobs by a keyword in the comment string of the jobs. `<allUsers>` specifies, using the value `True` or `False`, whether the jobs of all users should be considered—a value of `False` means that only the jobs owned by the requestor are considered. `GET_JOB_TOKEN` is needed to get any job slice, while `GET_OWN_JOB_TOKEN` is needed to get just the job slices belonging to the user making the call.

**getParentJobs(<scheduler>, <start>, <maxCount>, <parentID>, <allUsers>):** returns `parentJob` entities at the position `<start>` in the global list (sorted by parent job ID), but only up to `maxCount` items. That is, if the value of the parameter `<start>` is a number `n`, then jobs starting from the `n`th item in the global list are returned, up to `<maxCount>` times. `<parentID>` is a method to group jobs by a keyword in the comment string of the jobs. By default it has an empty value passed to it. If `<parentID>` is given a parent ID value, then the parent job is treated as owned by particular user if and only if all jobs with this tag (parent id) are submitted by that user. Setting `<allUsers>` specifies, using the value `True` or `False`, whether the jobs of all users should be considered—a value of `False` means that only the jobs owned by the requestor are considered. `GET_JOB_TOKEN` is needed to get any job slice, while `GET_OWN_JOB_TOKEN` is needed to get just the job slices belonging to the user making the call.

**requeueJob(<scheduler>, <JobID>):** requeues job and returns the result of this operation as a string. `REQUEUE_JOB_TOKEN` is needed to be able to requeue any job, while `REQUEUE_OWN_JOB_TOKEN` is needed to be able to requeue just the job belonging to the user making the call.

**holdJob(<scheduler>, <JobID>):** holds the job and returns the result of this operation as a string. `HOLD_JOB_TOKEN` is needed to be able to hold any job, while `HOLD_OWN_JOB_TOKEN` is needed to be able to hold just the job belonging to the user making the call.

**suspendJob**(*<scheduler>*, *<JobID>*): suspends the job and returns the result of this operation as a string. `SUSPEND_JOB_TOKEN` is needed to be able to suspend any job, while `SUSPEND_OWN_JOB_TOKEN` is needed to be able to suspend just the job belonging to the user making the call.

**resumeJob**(*<scheduler>*, *<JobID>*): resumes the job and returns the result of this operation as a string. `RESUME_JOB_TOKEN` is needed to be able to resume any job, while `RESUME_OWN_JOB_TOKEN` is needed to be able to resume just the job belonging to the user making the call.

**releaseJob**(*<scheduler>*, *<JobID>*): release the job and returns the result of this operation as a string. `RELEASE_JOB_TOKEN` is needed to be able to release any job, while `RELEASE_OWN_JOB_TOKEN` is needed to be able to release just the job belonging to the user making the call.

**updateJob**(*<scheduler>*, *<JobID>*): update the job and returns result of this operation as a string. `UPDATE_JOB_TOKEN` is needed to be able to update any job, while `UPDATE_OWN_JOB_TOKEN` is needed to be able to update just the job belonging to the user making the call.

**isNodeAllocatedForUser**(*<scheduler>*, *<username>*, *<hostname>*): returns true if at least one job owned by the user, as specified by the value of *<username>* allocates the host, as specified by the value of *<hostname>*.

Parent job is an entity introduced in Bright 7.3 and serves a goal of jobs clusterization. The jobs can be united by a tag surrounded by square brackets (for example "[*workflow1*]"). The tag is parsed by CMDaemon from the job comment line. The first entry of such a tag in the job comment is considered as the parent job ID. CMDaemon caches parent jobs, and an API client can request all the parent jobs or just some particular one. This allows the client to unite jobs by some user-defined property in a workflow, even if the workload manager does not support the workflow.

### 1.4.3 Queue Information And Management

For queue manipulation the following functions are used.

**getJobQueues**(*<scheduler>*): retrieves all `JobQueue` entities. Requires `GET_JOBQUEUE_TOKEN`.

**getJobQueue**(*<queuename>*): retrieves a particular `JobQueue` entity. Here *<queuename>* is a string. Requires `GET_JOBQUEUE_TOKEN`.

**getParallelEnvs**(*<scheduler>*): retrieves a list of `ParallelEnvironment` entities associated with a particular workload manager. Requires `GET_PE_TOKEN`.

**getJobQueueStates**(*<scheduler>*): retrieves a list of `JobQueueStat` entities. Requires `GET_JOBQUEUE_TOKEN`.

**updateJobQueue**(*<JobQueue>*, *<force>*): updates job queue properties defined by `JobQueue` entity. Parameter *<force>* is ignored for now. Requires `UPDATE_JOBQUEUE_TOKEN`.

**addJobQueue**(*<JobQueue>*, *<force>*): adds a new job queue to workload manager. If *<force>* has the value `True`, then the existing queue is recreated. Requires `ADD_JOBQUEUE_TOKEN`.

**removeJobQueue**(*<queueKey>*, *<force>*): removes queue by key. The key can be retrieved from the `JobQueue` entity requested by the `getJobQueue` method. Parameter *<force>* is ignored for now. Requires `UPDATE_JOBQUEUE_TOKEN`.

**drainNodes**(*<scheduler>*, *<queue>*, *<nodes>*, *<drain>*): drains nodes (as defined by a list of hostnames or uniqueKeys) or a particular queue (if supported by the workload manager) in the workload manager. If *<drain>* has the value 1, then the nodes will be drained, otherwise they are undrained. Returns a list `DrainResult` entities. Requires `DRAIN_TOKEN`.

**drainOverview**(*<scheduler>*, *<nodes>*): returns `DrainResult` entities with current drain state of the nodes. The nodes are defined by a list of hostnames or uniqueKeys. Requires `DRAIN_OVERVIEW_TOKEN`.



# 2

## Metric Collections

This chapter gives details on metric collections.

Section 10.4.4 of the *Administrator Manual* introduces metric collections, and describes how to add a metric collections script with `cmgui`.

This chapter covers how to add a metric collections script with `cmsh`. It also describes the output specification of a metric collections script, along with example outputs, so that a metric collections script can be made by the administrator.

### 2.1 Metric Collections Added Using `cmsh`

A metric collections script, `responsiveness`, is added in the `monitoring metrics` mode just like any other metric.

#### Example

```
[bright72]% monitoring metrics
[bright72->monitoring->metrics]% add responsiveness
[...[responsiveness]]% set command /cm/local/apps/cmd/scripts/metrics/s\
ample_responsiveness
[...*[responsiveness*]]% set classofmetric prototype; commit
```

For `classofmetric`, the value `prototype` is the class used to distinguish metric collections from normal metrics.

### 2.2 Metric Collections Initialization

When a metric collections script is added to `CMDaemon` for the first time, `CMDaemon` implicitly runs it with the `--initialize` flag. The output is used to define the collections table header structure. The structure is composed of the component metrics in the collections script, and the resulting structure is placed in the `CMDaemon` monitoring database. After the initialization step, data values can be added to the collections table during regular use of the script.

The displayed output of a metric collections script when using the `--initialize` flag is a list of available metrics and their parameter values. The format of each line in the list is:

```
metric <name[:parameter]> <unit> <class> "<description>" <cumulative> <min> <max>
```

where the items in the line are:

- `metric`: A bare word.
- `<name[:parameter]>`: The name of the metric, with for certain metrics a parameter value. For example, the metric `AlertLevel` can have the parameter `sum` assigned to it with the ":" character.

- *<unit>*: The unit of measurement that the metric uses.
- *<class>*: Any of:
  - ceph,
  - cgroups/blkio, cgroups/cpu, cgroups/memory, cgroups/network,
  - cluster,
  - cpu,
  - dellnss,
  - disk,
  - env,
  - gpu,
  - hadoop/dfs, hadoop/mapred, hadoop/metricsystem, hadoop/rpcdetailed, hadoop/ugu
  - internal,
  - lustre,
  - madoop/rpc,
  - mem,
  - misc
  - net,
  - openstack, openstack/api/allservices, openstack/api/compute, openstack/api/identity, openstack/api/image, openstack/api/network, openstack/api/orchestration, openstack/api/volume, openstack/hypervisors, openstack/vm/blockdevice, openstack/vm/cpu, openstack/vm/memory, openstack/vm/network, openstack/vm/other
  - os,
  - prototype,
  - workload.
- *<description>*: This can contain spaces, but should be enclosed with quotes.
- *<cumulative>*: Either *yes* or *no*. This indicates whether the metric increases monotonically (e.g., bytes received) or not (e.g., temperature).
- *<min>* and *<max>*: The minimum and maximum numeric values of this metric are determined dynamically based on the values so far.

### Example

```
[root@myheadnode metrics]# ./sample_responsiveness --initialize
metric util_sda % internal "Percentage of CPU time during which I/O
requests were issued to device sda" no 0 100
metric await_sda ms internal "The average time (in milliseconds) for
I/O requests issued to device sda to be served" no 0 500
```

## 2.3 Metric Collections Output During Regular Use

The output of a metric collection script without a flag is a list of outputs from the available metrics. The format of each line in the list is:

```
metric <name[:parameter]> <value> [infomessage]
```

where the parameters to the `metric` bare word are:

- `<name[:parameter]>`: The name of the metric, with optional parameter for some metrics.
- `<value>`: The numeric value of the measurement.
- `[infomessage]`: An optional infomessage.

### Example

```
[root@myheadnode metrics]# ./sample_responsiveness
metric await_sda 0.00
metric util_sda 0.00
[root@myheadnode metrics]#
```

If the output has more metrics than that suggested by when the `--initialize` flag is used, then the extra sampled data is discarded. If the output has fewer metrics, then the metrics are set to NaN (not a number) for the sample.

A metric or health check inside a metric collection appears as a check when viewing metrics or healthcheck lists. Attempting to remove such a check specifically using `cmsh` or `cmgui` only succeeds until the node is updated or rebooted. It is the metric collection itself that should have the check removed from within it, in order to remove the check from the list of checks permanently.

Setting a node that is UP to a CLOSED state, and then bringing it out of that state with the `open` command (section 5.5.4 of the *Administrator Manual*) also has CMDaemon run the metric collections script with the `--initialize` flag. This is useful for allowing CMDaemon to re-check what metrics in the collections can be sampled, and then re-configure them.

## 2.4 Metric Collections Error Handling

If the exit code of the script is 0, CMDaemon assumes that there is no error. So, with the `--initialize` flag active, despite no numeric value output, the script does not exit with an error.

If the exit code of the script is non-zero, the output of the script is assumed to be a diagnostic message and is passed to the head node. This shows up as an event in `cmsh` or `cmgui`.

For example, the `sample_ipmi` script uses the `ipmi-sensors` binary internally. Calling the binary directly returns an error code if the device has no IPMI configured. However, the `sample_ipmi` script in this case simply returns 0, and no output. The rationale here being that the administrator is aware of this situation and would not expect data from that IPMI anyway, let alone an error.

## 2.5 Metric Collections Consolidator Syntax

Metric collections can have a consolidator format defined per metric. The consolidator definition must be placed as an output in the line immediately preceding the corresponding metric initialization output line. The consolidator definition line can take the following forms:

```
consolidators default
consolidators none
consolidators CONSOLIDATORNAME FORMAT SPECIFICATION
```

The meanings of the texts after `consolidators` are as follows:

- **default:** The metrics follow the default consolidator names and interval values (section 10.7.4, page 424 of the *Administrator Manual*). That is, consolidator names take the value of Hour, Day, Week, while the interval values are the corresponding durations in seconds.
- **consolidators none:** No consolidation is done, only raw data values are collected for the metrics.
- **CONSOLIDATORNAME FORMAT SPECIFICATION:** This has the form:  
`<name:interval[:kind[:tablelength]]>...`
  - **name:** the consolidator name. A special feature here is that it can also define a new consolidator if the name does not already exist. Multiple consolidators can be defined in each consolidator definition line, with *name* separated from any preceding definition on the same line by a space.
  - **interval:** the duration in seconds, between consolidation, for the consolidator.
  - **kind:** an optional value of min, max, or average. By default it is average.
  - **tablelength:** an optional value for the length of the table, if *kind* has been specified. By default it is 1000.

## 2.6 Metric Collections Environment Variables

The following environment variables are available for a metric collection script, as well as for custom scripts, running from CMDaemon:

### On all devices:

**CMD\_HOSTNAME:** name of the device. For example:

```
CMD_HOSTNAME=myheadnode
```

### Only on non-node devices:

**CMD\_IP:** IP address of the device. For example:

```
CMD_IP=192.168.1.33
```

### Only on node devices:

Because these devices generally have multiple interfaces, the single environment variable **CMD\_IP** is often not enough to express these. Multiple interfaces are therefore represented by these environment variables:

- **CMD\_INTERFACES:** list of names of the interfaces attached to the node. For example:

```
CMD_INTERFACES=eth0 eth1 ipmi0 BOOTIF
```

- **CMD\_INTERFACE\_<interface>\_IP:** IP address of the interface with the name <interface>. For example:

```
CMD_INTERFACE_eth0_IP=10.141.255.254
CMD_INTERFACE_eth1_IP=0.0.0.0
```

- **CMD\_INTERFACE\_<interface>\_TYPE:** type of interface with the name <interface>. For example:

```
CMD_INTERFACE_eth1_TYPE=NetworkPhysicalInterface
CMD_INTERFACE_ipmi0_TYPE=NetworkBmcInterface
```

Possible values are:

- NetworkBmcInterface
  - NetworkPhysicalInterface
  - NetworkVLANInterface
  - NetworkAliasInterface
  - NetworkBondInterface
  - NetworkBridgeInterface
  - NetworkTunnelInterface
  - NetworkNetMapInterface
- CMD\_BMCUSERNAME: username for the BMC device at this node (if available).
  - CMD\_BMCPASSWORD: password for the BMC device at this node (if available).

To parse the above information to get the BMC IP address of the node for which this script samples, one could use (in Perl):

```
my $ip;
my $interfaces = $ENV{"CMD_INTERFACES"};
foreach my $interface ( split( " ", $interfaces ) ) {
    if( $ENV{"CMD_INTERFACE_" . $interface . "_TYPE"} eq
        "NetworkBmcInterface" ) {
        $ip = $ENV{"CMD_INTERFACE_" . $interface . "_IP"};
        last;
    }
}
# $ip holds the bmc ip
```

A list of environment variables available under the CMDaemon environment can be found by running a script under CMDaemon and exporting the environment variables to a file for viewing. For example, the `/cm/local/apps/cmd/scripts/healthchecks/testhealthcheck` script can be modified and run to sample on the head node, with the added line: `set>/tmp/environment`. The resulting file `/tmp/environment` that is generated as part of the healthcheck run then includes the `CMD_*` environment variables.

### Example

```
CMD_BMCPASSWORD
CMD_BMCUSERNAME
CMD_CLUSTERNAME
CMD_CMDSTARTEDTIME
CMD_DEVICE_TYPE
CMD_EXPORTS
CMD_FSEXPOR__SLASH_cm_SLASH_shared_AT_internalnet_ALLOWWRITE
CMD_FSEXPOR__SLASH_cm_SLASH_shared_AT_internalnet_HOSTS
CMD_FSEXPOR__SLASH_cm_SLASH_shared_AT_internalnet_PATH
CMD_FSEXPOR__SLASH_home_AT_internalnet_ALLOWWRITE
CMD_FSEXPOR__SLASH_home_AT_internalnet_HOSTS
CMD_FSEXPOR__SLASH_home_AT_internalnet_PATH
CMD_FSEXPOR__SLASH_var_SLASH_spool_SLASH_burn_AT_internalnet_ALLOWWRITE
CMD_FSEXPOR__SLASH_var_SLASH_spool_SLASH_burn_AT_internalnet_HOSTS
```

```
CMD_FSEXPORT__SLASH_var_SLASH_spool_SLASH_burn_AT_internalnet_PATH
CMD_HOSTNAME
CMD_INTERFACES
CMD_INTERFACE_eth0_IP
CMD_INTERFACE_eth0_MTU
CMD_INTERFACE_eth0_SPEED
CMD_INTERFACE_eth0_STARTIF
CMD_INTERFACE_eth0_TYPE
CMD_INTERFACE_eth1_IP
CMD_INTERFACE_eth1_MTU
CMD_INTERFACE_eth1_SPEED
CMD_INTERFACE_eth1_STARTIF
CMD_INTERFACE_eth1_TYPE
CMD_IP
CMD_MAC
CMD_METRICNAME
CMD_METRICPARAM
CMD_MOUNTS
CMD_NODEGROUPS
CMD_PARTITION
CMD_PORT
CMD_PROTOCOL
CMD_ROLES
CMD_SCRIPTTIMEOUT
CMD_STATUS
CMD_STATUS_CLOSED
CMD_STATUS_HEALTHCHECK_FAILED
CMD_STATUS_HEALTHCHECK_UNKNOWN
CMD_STATUS_MESSAGE
CMD_STATUS_RESTART_REQUIRED
CMD_STATUS_STATEFLAPPING
CMD_STATUS_USERMESSAGE
CMD_SYSINFO_SYSTEM_MANUFACTURER
CMD_SYSINFO_SYSTEM_NAME
CMD_USERDEFINED1
CMD_USERDEFINED2
```

## 2.7 Metric Collections Examples

Bright Cluster Manager has several scripts in the `/cm/local/apps/cmd/scripts/metrics` directory. Among them are the metric collections scripts `testmetriccollection` and `sample_responsiveness`. A glance through them while reading this chapter may be helpful.

## 2.8 Metric Collections On iDataPlex And Similar Units

IBM's iDataPlex is a specially engineered dual node rack unit. When the term iDataPlex is used in the following text in this section, it also implies any other dual node units that show similar behavior.

This section gives details on configuring an iDataPlex if IPMI metrics retrieval seems to skip most IPMI values from one of the nodes in the unit.

When carrying out metrics collections on an iDataPlex unit, Bright Cluster Manager should work without any issues. However, it may be that due to the special paired node design of an iDataPlex unit, most IPMI metrics of one member of the pair are undetectable by the `sample_ipmi` script sampling on that particular node. The missing IPMI metrics can instead be retrieved from the second member in the pair (along with the IPMI metrics of the second member).

The output may thus look something like:

### Example

```
[root@master01 ~]# cmsh
[master01]% device latestmetricdata node181 | grep Domain
Metric                               Value
-----
Domain_A_FP_Temp                     23
Domain_A_Temp1                       39
Domain_A_Temp2                       37
Domain_Avg_Power                     140
Domain_B_FP_Temp                     24
Domain_B_Temp1                       40
Domain_B_Temp2                       37
[master01]% device latestmetricdata node182 | grep Domain
Metric                               Value
-----
Domain_A_FP_Temp                     no data
Domain_A_Temp1                       no data
Domain_A_Temp2                       no data
Domain_Avg_Power                     170
Domain_B_FP_Temp                     no data
Domain_B_Temp1                       no data
Domain_B_Temp2                       no data
[master01]%
```

Because there are usually many iDataPlex units in the rack, the metrics retrieval response of each node pair in a unit should be checked for this behavior.

The issue can be dealt with by Bright Cluster Manager by modifying the configuration file for the `sample_ipmi` script in `/cm/local/apps/cmd/scripts/metrics/configfiles/sample_ipmi.conf`. Two parameters that can be configured there are `chassisContainsLeadNode` and `chassisContainsLeadNodeRegex`.

- Setting `chassisContainsLeadNode` to `on` forces the `sample_ipmi` script to treat the unit as an iDataPlex unit.

In particular:

- `auto` (recommended) means the unit is checked by the IPMI metric sample collection script for whether it behaves like an iDataPlex unit.
- `on` means the unit is treated as an iDataPlex node pair, with one node being a lead node that has all the IPMI metrics.
- `off` means the unit is treated as a non-iDataPlex node pair, with each node having normal behavior when retrieving IPMI metrics. This setting may need to be used in case the default value of `auto` ever falsely detects a node as part of an iDataPlex pair.

- The value of `chassisContainsLeadNodeRegex` can be set to a regular expression pattern that matches the system information pattern for the name, as obtained by `CMDaemon` for an iDataPlex unit (or similar clone unit). The pattern that it is matched against is the output of:

```
cmsh -c 'device ; sysinfo master | grep "^System Name"'
```

If the pattern matches, then the IPMI sample collection script assumes the unit behaves like an iDataPlex dual node pair. The missing IPMI data values are then looked for on the lead node.

The value of `chassisContainsLeadNodeRegex` is set to `iDataPlex` by default.





# 3

## Bright Cluster Manager JSON API

This chapter gives an alphabetical list of the JSON API services and entities available for Bright Cluster Manager. The API reference documentation for all available services and entities is available on the head node at:

`/cm/local/docs/cmd/json/index.html`.

Some examples of JSON use are given in section 3.3

### 3.1 Services

- 3.1.1 **auth**
- 3.1.2 **ceph**
- 3.1.3 **cert**
- 3.1.4 **cloud**
- 3.1.5 **device**
- 3.1.6 **gui**
- 3.1.7 **hadoop**
- 3.1.8 **job**
- 3.1.9 **keyvalue**
- 3.1.10 **lustre**
- 3.1.11 **main**
- 3.1.12 **mon**
- 3.1.13 **net**
- 3.1.14 **openstack**
- 3.1.15 **part**
- 3.1.16 **proc**
- 3.1.17 **prov**
- 3.1.18 **puppet**
- 3.1.19 **serv**
- 3.1.20 **session**
- 3.1.21 **test**
- 3.1.22 **ticket**
- 3.1.23 **user**

### 3.2 Entities

- 3.2.1 **BackupRole**

- 3.2.2 **BadEntityManagers**
- 3.2.3 **BasicResource**
- 3.2.4 **BillingHistory**
- 3.2.5 **BootRole**
- 3.2.6 **BurnConfig**
- 3.2.7 **BurnStatus**
- 3.2.8 **BurnTestStatus**
- 3.2.9 **Category**
- 3.2.10 **Ceph**
- 3.2.11 **CephMonitorRole**
- 3.2.12 **CephOSDAssociation**
- 3.2.13 **CephOSDPool**
- 3.2.14 **CephOSDRole**
- 3.2.15 **CephState**
- 3.2.16 **Certificate**
- 3.2.17 **CertificateRequest**
- 3.2.18 **Chassis**
- 3.2.19 **CloudDirectorRole**
- 3.2.20 **CloudGatewayRole**
- 3.2.21 **CloudImage**
- 3.2.22 **CloudJobDescription**
- 3.2.23 **CloudJobSubmissionStatus**
- 3.2.24 **CloudNode**
- 3.2.25 **CloudPrivateCloud**
- 3.2.26 **CloudProvider**
- 3.2.27 **CloudRegion**
- 3.2.28 **CloudSettings**
- 3.2.29 **CloudStaticIP**
- 3.2.30 **CloudStorageAction**
- 3.2.31 **CloudStorageNodeState**
- 3.2.32 **CloudType**
- 3.2.33 **CloudVirtualNetworkInterface**
- 3.2.34 **ClusterSetup**
- 3.2.35 **CMDaemonBackgroundTask**
- 3.2.36 **CMDaemonFailover**
- 3.2.37 **CMDaemonFailoverGroup**
- 3.2.38 **CMDaemonFailoverGroupStatus**
- 3.2.39 **CMDaemonFailoverPeer**
- 3.2.40 **CMDaemonFailoverStatus**
- 3.2.41 **CMDaemonStatus**
- 3.2.42 **CMService**
- 3.2.43 **CondorClientRole**
- 3.2.44 **CondorJob**
- 3.2.45 **CondorJobQueue**
- 3.2.46 **CondorJobQueueStat**
- 3.2.47 **CondorServerRole**
- 3.2.48 **ConfigSum**

- 3.2.49 ConfigurationOverlay
- 3.2.50 ConsolidatorConf
- 3.2.51 DatabaseRole
- 3.2.52 DellClustat
- 3.2.53 DellClustatGroup
- 3.2.54 DellClustatNode
- 3.2.55 DellDiskGroupInfo
- 3.2.56 DellPhysicalDiskDriveInfo
- 3.2.57 DellRAIDControllerInfo
- 3.2.58 DellSettings
- 3.2.59 DellSettingsFirmware
- 3.2.60 DellSettingsNicDevice
- 3.2.61 DellStorageInfo
- 3.2.62 DellVirtualDiskInfo
- 3.2.63 Device
- 3.2.64 DevStatus
- 3.2.65 DiskAssertion
- 3.2.66 DiskDevice
- 3.2.67 DiskInfo
- 3.2.68 DiskPartition
- 3.2.69 DiskRaid
- 3.2.70 DiskSetup
- 3.2.71 DiskVolume
- 3.2.72 DiskVolumeGroup
- 3.2.73 DrainResult
- 3.2.74 EBSattachAction
- 3.2.75 EBSdetachAction
- 3.2.76 EC2AMI
- 3.2.77 EC2AvailabilityZone
- 3.2.78 EC2EBSStorage
- 3.2.79 EC2EphemeralStorage
- 3.2.80 EC2PrivateCloud
- 3.2.81 EC2Provider
- 3.2.82 EC2Region
- 3.2.83 EC2RegionAMI
- 3.2.84 EC2Settings
- 3.2.85 EC2StaticIP
- 3.2.86 EC2Storage
- 3.2.87 EC2Type
- 3.2.88 EC2VirtualNetworkInterface
- 3.2.89 EntityManagersMD5
- 3.2.90 EthernetSwitch
- 3.2.91 FailoverRole
- 3.2.92 FakeData
- 3.2.93 FSExport
- 3.2.94 FSMount
- 3.2.95 FSPart

- 3.2.96 FSPartAssociation
- 3.2.97 FSPartBasicAssociation
- 3.2.98 FSPartProviderAssociation
- 3.2.99 GenericDevice
- 3.2.100 GenericResource
- 3.2.101 GPUInfo
- 3.2.102 GPUSettings
- 3.2.103 GpuUnit
- 3.2.104 GPUUnitInfo
- 3.2.105 GridEngineJob
- 3.2.106 GridEngineJobQueue
- 3.2.107 GridEngineJobQueueStat
- 3.2.108 GridEngineParallelEnvironment
- 3.2.109 Group
- 3.2.110 GuiCephOsdPoolInfo
- 3.2.111 GuiCephOverview
- 3.2.112 GuiCephPgslInfo
- 3.2.113 GuiClusterOverview
- 3.2.114 GuiCompleteOpenStackOverview
- 3.2.115 GuiDiskUsage
- 3.2.116 GuiGpuUnitOverview
- 3.2.117 GuiHadoopHDFSDetailHBase
- 3.2.118 GuiHadoopHDFSDetailHDFS
- 3.2.119 GuiHadoopHDFSDetailMapreduce
- 3.2.120 GuiHadoopHDFSDetailSpark
- 3.2.121 GuiHadoopHDFSDetailYarn
- 3.2.122 GuiHadoopHDFSDetailZooKeeper
- 3.2.123 GuiHadoopHDFSOverview
- 3.2.124 GuiJob
- 3.2.125 GuiNetSwitchStatus
- 3.2.126 GuiNetworkInterface
- 3.2.127 GuiNodeOverview
- 3.2.128 GuiNodeStatus
- 3.2.129 GuiOpenStackOverview
- 3.2.130 GuiOpenStackProjectOverview
- 3.2.131 GuiOpenStackTenantOverview
- 3.2.132 GuiPDUBank
- 3.2.133 GuiPDUOutlet
- 3.2.134 GuiWorkload
- 3.2.135 HadoopBaseConfiguration
- 3.2.136 HadoopDataNodeHDFSConfiguration
- 3.2.137 HadoopDataNodeRole
- 3.2.138 HadoopHBaseClientHDFSConfiguration
- 3.2.139 HadoopHBaseClientRole
- 3.2.140 HadoopHBaseServerHDFSConfiguration
- 3.2.141 HadoopHBaseServerRole
- 3.2.142 HadoopHDFS

- 3.2.143 HadoopHiveHDFSConfiguration
- 3.2.144 HadoopHiveRole
- 3.2.145 HadoopJob
- 3.2.146 HadoopJobQueue
- 3.2.147 HadoopJobQueueStat
- 3.2.148 HadoopJobTrackerHDFSConfiguration
- 3.2.149 HadoopJobTrackerRole
- 3.2.150 HadoopJournalHDFSConfiguration
- 3.2.151 HadoopJournalRole
- 3.2.152 HadoopKMServerHDFSConfiguration
- 3.2.153 HadoopKMServerRole
- 3.2.154 HadoopNameNodeHDFSConfiguration
- 3.2.155 HadoopNameNodeRole
- 3.2.156 HadoopNFSGatewayHDFSConfiguration
- 3.2.157 HadoopNFSGatewayRole
- 3.2.158 HadoopSecondaryNameNodeHDFSConfiguration
- 3.2.159 HadoopSecondaryNameNodeRole
- 3.2.160 HadoopSparkMasterHDFSConfiguration
- 3.2.161 HadoopSparkMasterRole
- 3.2.162 HadoopSparkWorkerHDFSConfiguration
- 3.2.163 HadoopSparkWorkerRole
- 3.2.164 HadoopSparkYARNHDFSConfiguration
- 3.2.165 HadoopSparkYARNRole
- 3.2.166 HadoopSqoopHDFSConfiguration
- 3.2.167 HadoopSqoopRole
- 3.2.168 HadoopTaskTrackerHDFSConfiguration
- 3.2.169 HadoopTaskTrackerRole
- 3.2.170 HadoopYARNClientHDFSConfiguration
- 3.2.171 HadoopYARNClientRole
- 3.2.172 HadoopYARNServerHDFSConfiguration
- 3.2.173 HadoopYARNServerRole
- 3.2.174 HadoopZooKeeperHDFSConfiguration
- 3.2.175 HadoopZooKeeperRole
- 3.2.176 HAProxyBackendInformation
- 3.2.177 HAProxyEntry
- 3.2.178 HAProxyEntryBind
- 3.2.179 HAProxyFrontendInformation
- 3.2.180 HAProxyNodeInformation
- 3.2.181 HAProxyRole
- 3.2.182 HAProxyServer
- 3.2.183 HAProxySharedSettings
- 3.2.184 HealthCheck
- 3.2.185 HeatMapData
- 3.2.186 IBSwitch
- 3.2.187 IniConfigFileCustomizationEntry
- 3.2.188 IPCPerm
- 3.2.189 IPResource

3.2.190 Job  
3.2.191 JobQueue  
3.2.192 JobQueuePlaceholder  
3.2.193 JobQueueStat  
3.2.194 KernelModule  
3.2.195 KeyValuePair  
3.2.196 LicenseInfo  
3.2.197 LicenseManagerRole  
3.2.198 LoginRole  
3.2.199 LSFBaseJob  
3.2.200 LSFBaseJobQueue  
3.2.201 LSFBaseJobQueueStat  
3.2.202 LSFClientRole  
3.2.203 LSFJob  
3.2.204 LSFJobQueue  
3.2.205 LSFJobQueueStat  
3.2.206 LSFServerRole  
3.2.207 LustreAlert  
3.2.208 LustreClientMount  
3.2.209 LustreFileSystem  
3.2.210 LustreFileSystemTarget  
3.2.211 LustreLog  
3.2.212 LustreOverview  
3.2.213 LustreServer  
3.2.214 LustreServerProfile  
3.2.215 LustreSettings  
3.2.216 LustreTargetMap  
3.2.217 LustreUser  
3.2.218 LustreVolume  
3.2.219 LustreVolumeNode  
3.2.220 MasterNode  
3.2.221 MasterRole  
3.2.222 MemcachedRole  
3.2.223 MemoryInfo  
3.2.224 Metric  
3.2.225 MetricPrmId  
3.2.226 MICHostRole  
3.2.227 MICInfo  
3.2.228 MICNode  
3.2.229 MICNodeCategory  
3.2.230 MICOverlay  
3.2.231 MICSettings  
3.2.232 MonConf  
3.2.233 MonGlobalConf  
3.2.234 MonHealthConf  
3.2.235 MonitoringRole  
3.2.236 MonMetricConf

- 3.2.237 **MsgQueue**
- 3.2.238 **MyrinetSwitch**
- 3.2.239 **Network**
- 3.2.240 **NetworkAliasInterface**
- 3.2.241 **NetworkBmcInterface**
- 3.2.242 **NetworkBondInterface**
- 3.2.243 **NetworkBridgeInterface**
- 3.2.244 **NetworkInterface**
- 3.2.245 **NetworkNetMapInterface**
- 3.2.246 **NetworkPhysicalInterface**
- 3.2.247 **NetworkTunnelInterface**
- 3.2.248 **NetworkVLANInterface**
- 3.2.249 **NewNode**
- 3.2.250 **NFSexportAction**
- 3.2.251 **NFSmountAction**
- 3.2.252 **NFSunexportAction**
- 3.2.253 **NFSunmountAction**
- 3.2.254 **Node**
- 3.2.255 **NodeCategory**
- 3.2.256 **NodeGroup**
- 3.2.257 **OpenLavaClientRole**
- 3.2.258 **OpenLavaJob**
- 3.2.259 **OpenLavaJobQueue**
- 3.2.260 **OpenLavaJobQueueStat**
- 3.2.261 **OpenLavaServerRole**
- 3.2.262 **OpenStack**
- 3.2.263 **OpenStackApiAgent**
- 3.2.264 **OpenStackApiDomain**
- 3.2.265 **OpenStackApiEndpoint**
- 3.2.266 **OpenStackApiEntity**
- 3.2.267 **OpenStackApiFlavor**
- 3.2.268 **OpenStackApiFloatingIP**
- 3.2.269 **OpenStackApiGroup**
- 3.2.270 **OpenStackApiHostAggregate**
- 3.2.271 **OpenStackApiHypervisor**
- 3.2.272 **OpenStackApiImage**
- 3.2.273 **OpenStackApiNetwork**
- 3.2.274 **OpenStackApiPort**
- 3.2.275 **OpenStackApiProject**
- 3.2.276 **OpenStackApiRole**
- 3.2.277 **OpenStackApiRoleAssignment**
- 3.2.278 **OpenStackApiRouter**
- 3.2.279 **OpenStackApiSecurityGroup**
- 3.2.280 **OpenStackApiServer**
- 3.2.281 **OpenStackApiService**
- 3.2.282 **OpenStackApiSubnet**
- 3.2.283 **OpenStackApiUser**

3.2.284 **OpenStackApiVolume**  
3.2.285 **OpenStackApiVolumeSnapshot**  
3.2.286 **OpenStackApiVolumeType**  
3.2.287 **OpenStackBareMetalApiRole**  
3.2.288 **OpenStackBareMetalConductorRole**  
3.2.289 **OpenStackBareMetalDiscoverdDNsmasqRole**  
3.2.290 **OpenStackBareMetalDiscoverdRole**  
3.2.291 **OpenStackBlockStorage**  
3.2.292 **OpenStackComputeApiEC2Role**  
3.2.293 **OpenStackComputeApiMetadataRole**  
3.2.294 **OpenStackComputeApiRole**  
3.2.295 **OpenStackComputeConductorRole**  
3.2.296 **OpenStackComputeRole**  
3.2.297 **OpenStackComputeSchedulerRole**  
3.2.298 **OpenStackComputeVNCProxyRole**  
3.2.299 **OpenStackConfigFileCustomization**  
3.2.300 **OpenStackDashboardRole**  
3.2.301 **OpenStackDataProcessingApiRole**  
3.2.302 **OpenStackDBaaSRole**  
3.2.303 **OpenStackDefaultUserRole**  
3.2.304 **OpenStackIdentityApiRole**  
3.2.305 **OpenStackImageApiRole**  
3.2.306 **OpenStackImageBackend**  
3.2.307 **OpenStackImageBackendCeph**  
3.2.308 **OpenStackImageBackendFS**  
3.2.309 **OpenStackImageRegistryRole**  
3.2.310 **OpenStackMessageQueueServerRole**  
3.2.311 **OpenStackNetworkApiRole**  
3.2.312 **OpenStackNetworkRole**  
3.2.313 **OpenStackNodeRole**  
3.2.314 **OpenStackNovalImageBackend**  
3.2.315 **OpenStackNovalImageBackendCeph**  
3.2.316 **OpenStackNovalImageBackendCow**  
3.2.317 **OpenStackObjectAccountRole**  
3.2.318 **OpenStackObjectApiRole**  
3.2.319 **OpenStackObjectContainerRole**  
3.2.320 **OpenStackObjectStoreRole**  
3.2.321 **OpenStackOrchestrationApiRole**  
3.2.322 **OpenStackOrchestrationRole**  
3.2.323 **OpenStackSettings**  
3.2.324 **OpenStackSettingsAdvanced**  
3.2.325 **OpenStackSettingsCMDaemonInteractions**  
3.2.326 **OpenStackSettingsCompute**  
3.2.327 **OpenStackSettingsCredentials**  
3.2.328 **OpenStackSettingsDatabase**  
3.2.329 **OpenStackSettingsLogging**  
3.2.330 **OpenStackSettingsNetworking**



- 3.2.331 **OpenStackSettingsPorts**
- 3.2.332 **OpenStackSettingsQuota**
- 3.2.333 **OpenStackSettingsUserPortal**
- 3.2.334 **OpenStackSettingsUsers**
- 3.2.335 **OpenStackStorage**
- 3.2.336 **OpenStackTelemetryAgentCentralRole**
- 3.2.337 **OpenStackTelemetryAgentComputeRole**
- 3.2.338 **OpenStackTelemetryAgentIpmiRole**
- 3.2.339 **OpenStackTelemetryAgentNotificationRole**
- 3.2.340 **OpenStackTelemetryAlarmEvaluatorRole**
- 3.2.341 **OpenStackTelemetryAlarmNotifierRole**
- 3.2.342 **OpenStackTelemetryApiRole**
- 3.2.343 **OpenStackTelemetryCollectorRole**
- 3.2.344 **OpenStackUserRole**
- 3.2.345 **OpenStackUserSettings**
- 3.2.346 **OpenStackVolumeApiRole**
- 3.2.347 **OpenStackVolumeBackend**
- 3.2.348 **OpenStackVolumeBackendCeph**
- 3.2.349 **OpenStackVolumeBackendNFS**
- 3.2.350 **OpenStackVolumeBackupBackend**
- 3.2.351 **OpenStackVolumeBackupBackendCeph**
- 3.2.352 **OpenStackVolumeBackupRole**
- 3.2.353 **OpenStackVolumeRole**
- 3.2.354 **OpenStackVolumeSchedulerRole**
- 3.2.355 **OsapiPortIP**
- 3.2.356 **OsapiSecurityGroupRule**
- 3.2.357 **OsapiSubnetAllocationPool**
- 3.2.358 **OSService**
- 3.2.359 **OSServiceArray**
- 3.2.360 **OSServiceConfig**
- 3.2.361 **Partition**
- 3.2.362 **PBSJob**
- 3.2.363 **PBSJobQueue**
- 3.2.364 **PBSJobQueueStat**
- 3.2.365 **PbsProClientRole**
- 3.2.366 **PbsProJob**
- 3.2.367 **PbsProJobQueue**
- 3.2.368 **PbsProJobQueueStat**
- 3.2.369 **PbsProServerRole**
- 3.2.370 **PDUPort**
- 3.2.371 **PhysicalNode**
- 3.2.372 **PowerDistributionUnit**
- 3.2.373 **PowerStatus**
- 3.2.374 **Process**
- 3.2.375 **Processor**
- 3.2.376 **Profile**
- 3.2.377 **ProgramRunnerInput**

**3.2.378 ProgramRunnerKill**  
**3.2.379 ProgramRunnerOutput**  
**3.2.380 ProgramRunnerStatus**  
**3.2.381 ProvisioningNodeStatus**  
**3.2.382 ProvisioningProcessorJob**  
**3.2.383 ProvisioningRequestStatus**  
**3.2.384 ProvisioningRole**  
**3.2.385 ProvisioningStatus**  
**3.2.386 Puppet**  
**3.2.387 PuppetApplyResult**  
**3.2.388 PuppetApplySession**  
**3.2.389 PuppetClass**  
**3.2.390 PuppetRole**  
**3.2.391 PuppetRunInfo**  
**3.2.392 Rack**  
**3.2.393 RackSensor**  
**3.2.394 RadosGatewayRole**  
**3.2.395 RateElem**  
**3.2.396 ReadMonDataId**  
**3.2.397 ReadMonDataOutput**  
**3.2.398 RemoteMonConf**  
**3.2.399 RemoteMonMetricConf**  
**3.2.400 RemoteNodeInstallerInteraction**  
**3.2.401 RemoteSetupExecution**  
**3.2.402 RemoteThreshold**  
**3.2.403 ResourcePool**  
**3.2.404 ResourcePoolStatus**  
**3.2.405 Role**  
**3.2.406 RunJobAction**  
**3.2.407 S3DataDownload**  
**3.2.408 S3DataUpload**  
**3.2.409 S3ResultsDownload**  
**3.2.410 S3ResultsUpload**  
**3.2.411 S3Transfer**  
**3.2.412 Semaphore**  
**3.2.413 Sensor**  
**3.2.414 Session**  
**3.2.415 SGEClientRole**  
**3.2.416 SGEJob**  
**3.2.417 SGEJobQueue**  
**3.2.418 SGEJobQueueStat**  
**3.2.419 SGEParallelEnvironment**  
**3.2.420 SGEServerRole**  
**3.2.421 SharedMemory**  
**3.2.422 SlaveMonotonicElem**  
**3.2.423 SlaveMonSnapshot**  
**3.2.424 SlaveNode**

---

3.2.425 SlaveRateElem  
3.2.426 SlurmClientRole  
3.2.427 SlurmJob  
3.2.428 SlurmJobQueue  
3.2.429 SlurmJobQueueStat  
3.2.430 SlurmServerRole  
3.2.431 SoftwareImage  
3.2.432 SoftwareImageProxy  
3.2.433 StartStorageNodeAction  
3.2.434 StateElem  
3.2.435 StaticRoute  
3.2.436 StatisticData  
3.2.437 StopStorageNodeAction  
3.2.438 StorageNodePolicy  
3.2.439 StorageRole  
3.2.440 StringListObject  
3.2.441 SubnetManagerRole  
3.2.442 Switch  
3.2.443 SwitchPort  
3.2.444 SysInfoCollector  
3.2.445 SysMonotonicWithId  
3.2.446 SysRateWithId  
3.2.447 ThreshAction  
3.2.448 ThreshActionConf  
3.2.449 Threshold  
3.2.450 Ticket  
3.2.451 TorqueClientRole  
3.2.452 TorqueJob  
3.2.453 TorqueJobQueue  
3.2.454 TorqueJobQueueStat  
3.2.455 TorqueServerRole  
3.2.456 UCSAdaptorEthCompQueueProfile  
3.2.457 UCSAdaptorEthGenProfile  
3.2.458 UCSAdaptorEthInterruptProfile  
3.2.459 UCSAdaptorEthOffloadProfile  
3.2.460 UCSAdaptorEthRecvQueueProfile  
3.2.461 UCSAdaptorEthUSNICProfile  
3.2.462 UCSAdaptorEthWorkQueueProfile  
3.2.463 UCSAdaptorExtEthIf  
3.2.464 UCSAdaptorExtIpv6RssHashProfile  
3.2.465 UCSAdaptorFcCdbWorkQueueProfile  
3.2.466 UCSAdaptorFcErrorRecoveryProfile  
3.2.467 UCSAdaptorFcGenProfile  
3.2.468 UCSAdaptorFcInterruptProfile  
3.2.469 UCSAdaptorFcPortFLogiProfile  
3.2.470 UCSAdaptorFcPortPLogiProfile  
3.2.471 UCSAdaptorFcPortProfile

3.2.472 UCSAdaptorFcRecvQueueProfile  
3.2.473 UCSAdaptorFcWorkQueueProfile  
3.2.474 UCSAdaptorHostEthIf  
3.2.475 UCSAdaptorHostFclIf  
3.2.476 UCSAdaptorIpV4RssHashProfile  
3.2.477 UCSAdaptorIpV6RssHashProfile  
3.2.478 UCSAdaptorPortProfiles  
3.2.479 UCSAdaptorRssProfile  
3.2.480 UCSBase  
3.2.481 UCSBiosBootDev  
3.2.482 UCSBiosBootDevGrp  
3.2.483 UCSBiosSettings  
3.2.484 UCSBiosVfAdjacentCacheLinePrefetch  
3.2.485 UCSBiosVfAltitude  
3.2.486 UCSBiosVfASPMSupport  
3.2.487 UCSBiosVfConsoleRedirection  
3.2.488 UCSBiosVfCoreMultiProcessing  
3.2.489 UCSBiosVfCPUEnergyPerformance  
3.2.490 UCSBiosVfCPUFrequencyFloor  
3.2.491 UCSBiosVfCPUPerformance  
3.2.492 UCSBiosVfCPUPowerManagement  
3.2.493 UCSBiosVfDCUPrefetch  
3.2.494 UCSBiosVfDemandScrub  
3.2.495 UCSBiosVfDirectCacheAccess  
3.2.496 UCSBiosVfDRAMClockThrottling  
3.2.497 UCSBiosVfDramRefreshRate  
3.2.498 UCSBiosVfEnhancedIntelSpeedStepTech  
3.2.499 UCSBiosVfExecuteDisableBit  
3.2.500 UCSBiosVfFRB2Enable  
3.2.501 UCSBiosVfHardwarePrefetch  
3.2.502 UCSBiosVfIntelHyperThreadingTech  
3.2.503 UCSBiosVfIntelTurboBoostTech  
3.2.504 UCSBiosVfIntelVirtualizationTechnology  
3.2.505 UCSBiosVfIntelVTFForDirectedIO  
3.2.506 UCSBiosVfLegacyUSBSupport  
3.2.507 UCSBiosVfLOMPortOptionROM  
3.2.508 UCSBiosVfLvDIMMSupport  
3.2.509 UCSBiosVfMemoryInterleave  
3.2.510 UCSBiosVfMemoryMappedIOAbove4GB  
3.2.511 UCSBiosVfNUMAOptimized  
3.2.512 UCSBiosVfOnboardStorage  
3.2.513 UCSBiosVfOnboardStorageSWStack  
3.2.514 UCSBiosVfOSBootWatchdogTimer  
3.2.515 UCSBiosVfOSBootWatchdogTimerPolicy  
3.2.516 UCSBiosVfOSBootWatchdogTimerTimeout  
3.2.517 UCSBiosVfPatrolScrub  
3.2.518 UCSBiosVfPCIOptionROMs

3.2.519 UCSBiosVfPCISlotOptionROMEnable  
3.2.520 UCSBiosVfProcessorC1E  
3.2.521 UCSBiosVfProcessorC6Report  
3.2.522 UCSBiosVfPStateCoordType  
3.2.523 UCSBiosVfQPIConfig  
3.2.524 UCSBiosVfSelectMemoryRASConfiguration  
3.2.525 UCSBiosVfTPMSupport  
3.2.526 UCSBiosVfUCSMBootOrderRuleControl  
3.2.527 UCSBiosVfUSBEmulation  
3.2.528 UCSBiosVfUSBPortsConfig  
3.2.529 UCSBiosVfVgaPriority  
3.2.530 UCSCommNtpProvider  
3.2.531 UCSCommSyslog  
3.2.532 UCSCommSyslogClient  
3.2.533 UCSEquipmentIndicatorLed  
3.2.534 UCSEquipmentLocatorLed  
3.2.535 UCSFaultInst  
3.2.536 UCSFirmwareRunning  
3.2.537 UCSInfo  
3.2.538 UCSLogs  
3.2.539 UCSSLsbootDef  
3.2.540 UCSSLsbootEfi  
3.2.541 UCSSLsbootLan  
3.2.542 UCSSLsbootStorage  
3.2.543 UCSSLsbootVirtualMedia  
3.2.544 UCSStatus  
3.2.545 UGECgroupsSettings  
3.2.546 UGEClientRole  
3.2.547 UGEJob  
3.2.548 UGEJobQueue  
3.2.549 UGEJobQueueStat  
3.2.550 UGEParallelEnvironment  
3.2.551 UGEServerRole  
3.2.552 User  
3.2.553 Validation  
3.2.554 VersionInfo  
3.2.555 VirtualNode  
3.2.556 VirtualNodeSettings  
3.2.557 VirtualSMPNode  
3.2.558 VScaleMPSettings  
3.2.559 VsmptSettings  
3.2.560 WillChange  
3.2.561 XeonPhiSettings

### 3.3 JSON Examples

complete.sh

```
#!/bin/bash

URL=https://localhost:2081/json/
user=koen
pass=koen

echo "==== login ====="
curl -c curl.cookie.txt -i -k -X POST -d '{"service":"login", \
"username":"koen", "password":"' $pass'"}' $URL; echo
echo "==== master ====="
curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"cm\
device", "call":"getNode", "arg":"master"}' $URL; echo
echo "==== logout ====="
curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"lo\
gout"}' $URL; echo
echo "==== denied ====="
curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"cm\
device", "call":"getNode", "arg":"master"}' $URL; echo
rm -f curl.cookie.txt

echo "==== cert ====="
curl --cert $HOME/.cm/admin.pem --key $HOME/.cm/admin.key -i -k \
-X POST -d '{"service":"cmdevice", "call":"getNode", "arg":"master\
"}' $URL; echo
```

### curl.sh

```
#!/bin/bash

source url
if [ -z "$1" ]; then
    pass=koen
else
    pass=$1
fi
read -p "pass: " -s -a $pass

curl -c curl.cookie.txt -i -k -X POST -d '{"service":"login", \
"username":"koen", "password":"' $pass'"}' $URL

# curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"\
cmsession", "call":"getLastEvents", "args":[0,256]}' $URL

curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"cm\
main", "call":"getProfile"}' $URL
curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"cm\
main", "call":"getSubjectName"}' $URL
```

### devices.sh

```
#!/bin/bash
source url

if [ "$1" == "gzip" ]; then
```

```

    wget --load-cookies cookie.txt --header='Accept-Encoding: gzip\
' --no-check-certificate --server-response -qO- $URL --post-dat\
a='{"service":"cmdevice","call":"getDevices"}'
else
    wget --load-cookies cookie.txt --no-check-certificate --server\
-response -qO- $URL --post-data='{"service":"cmdevice","call":"g\
etDevices"}'
fi

```

### loadone.sh

```

#!/bin/bash
source url

# not perfect but gets the job done
function jsonval {
    temp=`echo $json | sed 's/\\\\\\\\/\\/g' | sed 's/[{}]/g' | awk\
-v k="text" '{n=split($0,a,","); for (i=1; i<=n; i++) print a[i\
]}' | sed 's/\"\\:\\"/\\/g' | sed 's/[\\,]/ /g' | sed 's/\"/\\/g' | g\
rep -w $prop`
    r=$(echo ${temp##*|} | tr ']' ' ' | tr ' ' '\n' | cut -d: -f2 \
| sort -n)
    echo $(echo $r | cut -d' ' -f 1)
}

prop='uniqueKey'

node=master
json=`wget --load-cookies cookie.txt --no-check-certificate --se\
rver-response -qO- $URL --post-data='{"service":"cmdevice","call\
":"getDevice","arg1":"' $node' }'`
nkey=$(jsonval)
if [ -z $nkey ]; then
    echo $json
    exit 1
fi
echo "$node.uniqueKey = $nkey"

json=`wget --load-cookies cookie.txt --no-check-certificate --se\
rver-response -qO- $URL --post-data='{"service":"cmmon","call": "\
getMetric","arg1":"loadOne"}'`
mkey=$(jsonval)
echo "loadone.uniqueKey = $mkey"

now=$(date +%s)
day=$((now-86400))

# echo -----
# wget --load-cookies cookie.txt --no-check-certificate --server\
-response -qO- $URL \
# --post-data='{"service":"cmmon","call":"readDataByIntervalNu\
m",
# "readMonDataIdArray":[{"devId":"' $nkey', "metric\
Id":"' $mkey',

```

```

#                                     "begTime": '$day', "endTi\
me": '$now' }},
#                                     "intervalNum": 0}'
#
# echo
echo -----
wget --load-cookies cookie.txt --no-check-certificate --server-r\
esponse -qO- $URL \
  --post-data='{ "service": "cmmon", "call": "readDataByIntervalNum",
                "args": [{"baseType": "ReadMonDataId", "uniqueKey"\
: 0, "modified": false, "toBeRemoved": false, "childType": "",
                        "devId": '$nkey', "metricId": '$mkey',
                        "begTime": '$day', "endTime": '$now' }], 0}'

# echo
# echo -----
# data='{ "service": "cmmon", "call": "readDataByIntervalNum",
#        "args": [{"baseType": "ReadMonDataId", "uniqueKe\
y": 0, "modified": false, "toBeRemoved": false, "childType": "",
#                "devId": '$nkey', "metricId": '$mkey',
#                "begTime": '$day', "endTime": '$now' }], \
0}'
# rm loadone.txt.gz
# echo $data > loadone.txt
# gzip -n loadone.txt
# len=$(wc -c loadone.txt.gz | cut -d" " -f1)
# wget --load-cookies cookie.txt --no-check-certificate --header\
"Content-Length: $len" --header 'Content-Encoding: gzip' --serv\
er-response -O- $URL \
#   --post-file=loadone.txt.gz

```

### login.sh

```

#!/bin/bash
source url
user=$USER
pass=$user
wget --keep-session-cookies --save-cookies cookie.txt --no-check\
-certificate --server-response -qO- $URL \
  --post-data='{ "service": "login", "username": "'$user'", "passwor\
d": "'$pass' }'
echo

```

### logout.sh

```

#!/bin/bash
source url
wget --load-cookies cookie.txt --no-check-certificate --server-r\
esponse -qO- $URL --post-data='{ "service": "logout" }'
rm cookie.txt
echo

```

### node001.sh

```

#!/bin/bash
source url

```



```
if [ -z "$1" ]; then
    node=node001
else
    node=$1
fi

wget --load-cookies cookie.txt --no-check-certificate --server-r\
esponse -qO- $URL --post-data="{\"service\":\"cmdevice\",\"call\":\"get\
Device\",\"arg1\":\"$node\"}"'
```