

Bright Cluster Manager 8.1

Developer Manual

Revision: c2da708

Date: Mon Aug 17 2020



©2020 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. PBS Professional, PBS Pro, and Green Provisioning are trademarks of Altair Engineering, 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	xxi
0.2 About The Manuals In General	xxi
0.3 Getting Administrator-Level Support	xxii
0.4 Getting Developer-Level Support	xxii
0.5 Getting Professional Services	xxii
1 Bright Cluster Manager Python API	1
1.1 Installation	1
1.1.1 Linux Clients	1
1.2 Examples	1
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	3
1.3.3 Creating New Objects	3
1.3.4 List Of Objects	4
1.3.5 Useful Methods	6
1.3.6 Useful Example Program	7
1.4 The Workload Management API	8
1.4.1 Job Submission	8
1.4.2 Job Information And Management	11
1.4.3 Queue Information And Management	12
2 Monitoring Data Producers	15
2.1 Measurables	15
2.2 Measurables Classes	15
2.3 Metric Monitoring Data Producers	15
2.4 Health Check Monitoring Data Producers	16
2.5 Collection Monitoring Data Producers	17
2.6 Perpetual Monitoring Data Producers	18
2.7 Prometheus Monitoring Data Producers	19
2.8 Node Execution Filters	19
2.9 Execution Multiplexers	20
2.10 Monitoring Resources	21
2.11 Collection Monitoring Data Producers With Filter And Multiplexer	21
2.12 Collection Monitoring Data Producers For Standalone Entities	22

3	Monitoring Actions	25
3.1	Actions And Triggers	25
3.2	Time Restrictions	26
3.2.1	Time Restriction Syntax In BNF Notation	26
3.3	CMDaemon Environment Variables	26
3.3.1	Standard Environment Variables Available In Action Scripts	26
3.3.2	Extended Environment Variables Available To Action Scripts	28
4	Bright Cluster Manager JSON API	41
4.1	Services	41
4.1.1	auth	41
4.1.2	ceph	41
4.1.3	cert	41
4.1.4	cloud	41
4.1.5	device	41
4.1.6	etcd	41
4.1.7	gui	41
4.1.8	hadoop	41
4.1.9	job	41
4.1.10	keyvalue	41
4.1.11	kube	41
4.1.12	lustre	41
4.1.13	main	41
4.1.14	mesos	41
4.1.15	mon	41
4.1.16	net	41
4.1.17	openstack	41
4.1.18	part	41
4.1.19	proc	41
4.1.20	prov	41
4.1.21	serv	41
4.1.22	session	41
4.1.23	test	41
4.1.24	ticket	42
4.1.25	user	42
4.1.26	zookeeper	42
4.2	Entities	42
4.2.1	AMDGPUSettings	42
4.2.2	AzureDataDisk	42
4.2.3	AzureDisk	42
4.2.4	AzureIntermediateStorage	42
4.2.5	AzureLocation	42
4.2.6	AzureManagedDiskParameters	42
4.2.7	AzureOSDisk	42
4.2.8	AzureProvider	42
4.2.9	AzureSettings	42
4.2.10	AzureVMSize	42

4.2.11	BadEntityManagers	42
4.2.12	BasicResource	42
4.2.13	BeeGFSAdmonRole	42
4.2.14	BeeGFSClientRole	42
4.2.15	BeeGFSManagementRole	42
4.2.16	BeeGFSMetadataRole	42
4.2.17	BeeGFSStorageRole	42
4.2.18	BigDataAdditionalTool	42
4.2.19	BigDataAdvancedSettings	42
4.2.20	BigDataCassandra	42
4.2.21	BigDataFileSystemSettings	42
4.2.22	BigDataJobManagementSettings	42
4.2.23	BigDataLoggingSettings	42
4.2.24	BigDataSecurity	42
4.2.25	BigDataSpark	42
4.2.26	BillingHistory	42
4.2.27	BMCSettings	42
4.2.28	BootRole	42
4.2.29	BurnConfig	42
4.2.30	BurnStatus	42
4.2.31	BurnTestStatus	42
4.2.32	Category	42
4.2.33	Ceph	42
4.2.34	CephMDSRole	42
4.2.35	CephMGRRole	42
4.2.36	CephMonitorRole	42
4.2.37	CephOSDBlueStoreConfig	42
4.2.38	CephOSDConfig	42
4.2.39	CephOSDFileStoreConfig	42
4.2.40	CephOSDLegacyConfig	42
4.2.41	CephOSDPool	42
4.2.42	CephOSDRole	42
4.2.43	CephState	43
4.2.44	Certificate	43
4.2.45	CertificateRequest	43
4.2.46	CertificateSubjectName	43
4.2.47	Cgroup	43
4.2.48	CgroupController	43
4.2.49	CgroupControllerBlkio	43
4.2.50	CgroupControllerCpu	43
4.2.51	CgroupControllerCpuacct	43
4.2.52	CgroupControllerCpuset	43
4.2.53	CgroupControllerDevices	43
4.2.54	CgroupControllerFreezer	43
4.2.55	CgroupControllerHugetlb	43
4.2.56	CgroupControllerMemory	43

4.2.57	CgroupControllerNetcls	43
4.2.58	CgroupControllerNetprio	43
4.2.59	CgroupControllerNs	43
4.2.60	CgroupControllerPerf	43
4.2.61	CgroupRule	43
4.2.62	CgroupSupervisorRole	43
4.2.63	Chassis	43
4.2.64	ChronosRole	43
4.2.65	ClientUserData	43
4.2.66	CloudDirectorRole	43
4.2.67	CloudGatewayRole	43
4.2.68	CloudImage	43
4.2.69	CloudJobDescription	43
4.2.70	CloudJobSubmissionStatus	43
4.2.71	CloudNode	43
4.2.72	CloudProvider	43
4.2.73	CloudRegion	43
4.2.74	CloudSettings	43
4.2.75	CloudStaticIP	43
4.2.76	CloudStorageActionData	43
4.2.77	CloudStorageNodeState	43
4.2.78	CloudType	43
4.2.79	ClusterSetup	43
4.2.80	CMDaemonBackgroundTask	43
4.2.81	CMDaemonFailover	43
4.2.82	CMDaemonFailoverGroup	43
4.2.83	CMDaemonFailoverGroupStatus	43
4.2.84	CMDaemonFailoverPeer	43
4.2.85	CMDaemonFailoverStatus	43
4.2.86	CMDaemonStatus	43
4.2.87	CMService	43
4.2.88	CMSubConfig	43
4.2.89	CMSubIntermediateStorage	43
4.2.90	ConfigFileVersion	44
4.2.91	ConfigSum	44
4.2.92	ConfigurationOverlay	44
4.2.93	Consolidator	44
4.2.94	ContainerdHostRole	44
4.2.95	ContainerInfo	44
4.2.96	CustomizationEntry	44
4.2.97	CustomizationFile	44
4.2.98	DellClustat	44
4.2.99	DellClustatGroup	44
4.2.100	DellClustatNode	44
4.2.101	DellDiskGroupInfo	44
4.2.102	DellPhysicalDiskDriveInfo	44

4.2.103 DellRAIDControllerInfo	44
4.2.104 DellSettings	44
4.2.105 DellSettingsFirmware	44
4.2.106 DellSettingsNicDevice	44
4.2.107 DellStorageInfo	44
4.2.108 DellVirtualDiskInfo	44
4.2.109 Device	44
4.2.110 DevStatus	44
4.2.111 DIGITSRole	44
4.2.112 DiskAssertion	44
4.2.113 DiskDevice	44
4.2.114 DiskInfo	44
4.2.115 DiskPartition	44
4.2.116 DiskRaid	44
4.2.117 DiskSetup	44
4.2.118 DiskVolume	44
4.2.119 DiskVolumeGroup	44
4.2.120 DockerHostRole	44
4.2.121 DockerRegistryFilesystemStorageDriver	44
4.2.122 DockerRegistryInmemoryStorageDriver	44
4.2.123 DockerRegistryRole	44
4.2.124 DockerRegistryStorageDriver	44
4.2.125 DockerStorageAufsBackend	44
4.2.126 DockerStorageBackend	44
4.2.127 DockerStorageDeviceMapperBackend	44
4.2.128 DockerStorageOverlay2Backend	44
4.2.129 DrainAction	44
4.2.130 DrainResult	44
4.2.131 EC2AMI	44
4.2.132 EC2AvailabilityZone	44
4.2.133 EC2EBSStorage	44
4.2.134 EC2EphemeralStorage	44
4.2.135 EC2Provider	44
4.2.136 EC2Region	44
4.2.137 EC2RegionAMI	45
4.2.138 EC2Settings	45
4.2.139 EC2StaticIP	45
4.2.140 EC2Storage	45
4.2.141 EC2Type	45
4.2.142 EC2VPC	45
4.2.143 ElasticSearchRole	45
4.2.144 EntityManagersMD5	45
4.2.145 EtcCluster	45
4.2.146 EtcHostRole	45
4.2.147 EthernetSwitch	45
4.2.148 FailoverRole	45

4.2.149 FakeJob	45
4.2.150 FakeJobQueue	45
4.2.151 FakeJobQueueStat	45
4.2.152 FakeWlmClientRole	45
4.2.153 FakeWlmServerRole	45
4.2.154 FileInfo	45
4.2.155 FileSyncConfig	45
4.2.156 FileSyncStatus	45
4.2.157 FlannelConfigurationRole	45
4.2.158 FlannelHostRole	45
4.2.159 FlannelNetworkingBackend	45
4.2.160 FlannelNetworkingUdpBackend	45
4.2.161 FlannelNetworkingVxLanBackend	45
4.2.162 FSExport	45
4.2.163 FSMount	45
4.2.164 FSPart	45
4.2.165 FSPartAssociation	45
4.2.166 FSPartBasicAssociation	45
4.2.167 FSPartProviderAssociation	45
4.2.168 GaleraRole	45
4.2.169 GenericDevice	45
4.2.170 GenericResource	45
4.2.171 GenericRole	45
4.2.172 GenericRoleConfiguration	45
4.2.173 GenericRoleEnvironment	45
4.2.174 GenericRoleGeneratedConfiguration	45
4.2.175 GenericRoleStaticConfiguration	45
4.2.176 GenericRoleSymlinkConfiguration	45
4.2.177 GenericRoleTemplatedConfiguration	45
4.2.178 GPUInfo	45
4.2.179 GPUSettings	45
4.2.180 GpuUnit	45
4.2.181 GPUUnitInfo	45
4.2.182 GridEngineJob	45
4.2.183 GridEngineJobQueue	45
4.2.184 GridEngineJobQueueStat	46
4.2.185 GridEngineParallelEnvironment	46
4.2.186 Group	46
4.2.187 GuiCephOsdPoolInfo	46
4.2.188 GuiCephOverview	46
4.2.189 GuiCephPgsInfo	46
4.2.190 GuiClusterOverview	46
4.2.191 GuiCompleteOpenStackOverview	46
4.2.192 GuiDiskUsage	46
4.2.193 GuiGpuUnitOverview	46
4.2.194 GuiHadoopHDFSDetailHBase	46

4.2.195 GuiHadoopHDFSDetailHDFS	46
4.2.196 GuiHadoopHDFSDetailMapreduce	46
4.2.197 GuiHadoopHDFSDetailSpark	46
4.2.198 GuiHadoopHDFSDetailYarn	46
4.2.199 GuiHadoopHDFSDetailZooKeeper	46
4.2.200 GuiHadoopHDFSOverview	46
4.2.201 GuiJob	46
4.2.202 GuiKubeClusterOverview	46
4.2.203 GuiNetworkInterface	46
4.2.204 GuiNodeOverview	46
4.2.205 GuiNodeStatus	46
4.2.206 GuiOpenStackOverview	46
4.2.207 GuiOpenStackProjectOverview	46
4.2.208 GuiOpenStackTenantOverview	46
4.2.209 GuiPDUBank	46
4.2.210 GuiPDUOutlet	46
4.2.211 GuiPDUOverview	46
4.2.212 GuiSwitchOverview	46
4.2.213 GuiSwitchPort	46
4.2.214 GuiWorkload	46
4.2.215 HadoopAccumuloMasterHDFSConfiguration	46
4.2.216 HadoopAccumuloMasterRole	46
4.2.217 HadoopAccumuloTabletHDFSConfiguration	46
4.2.218 HadoopAccumuloTabletRole	46
4.2.219 HadoopAlluxioMasterHDFSConfiguration	46
4.2.220 HadoopAlluxioMasterRole	46
4.2.221 HadoopAlluxioWorkerHDFSConfiguration	46
4.2.222 HadoopAlluxioWorkerRole	46
4.2.223 HadoopBaseConfiguration	46
4.2.224 HadoopCassandraHDFSConfiguration	46
4.2.225 HadoopCassandraRole	46
4.2.226 HadoopDataNodeHDFSConfiguration	46
4.2.227 HadoopDataNodeRole	46
4.2.228 HadoopDrillHDFSConfiguration	46
4.2.229 HadoopDrillRole	46
4.2.230 HadoopFlinkJobManagerHDFSConfiguration	46
4.2.231 HadoopFlinkJobManagerRole	47
4.2.232 HadoopFlinkTaskManagerHDFSConfiguration	47
4.2.233 HadoopFlinkTaskManagerRole	47
4.2.234 HadoopHBaseClientHDFSConfiguration	47
4.2.235 HadoopHBaseClientRole	47
4.2.236 HadoopHBaseServerHDFSConfiguration	47
4.2.237 HadoopHBaseServerRole	47
4.2.238 HadoopHDFS	47
4.2.239 HadoopHiveHDFSConfiguration	47
4.2.240 HadoopHiveRole	47

4.2.241 HadoopJob	47
4.2.242 HadoopJobQueue	47
4.2.243 HadoopJobQueueStat	47
4.2.244 HadoopJobTrackerHDFSConfiguration	47
4.2.245 HadoopJobTrackerRole	47
4.2.246 HadoopJournalHDFSConfiguration	47
4.2.247 HadoopJournalRole	47
4.2.248 HadoopKafkaServerHDFSConfiguration	47
4.2.249 HadoopKafkaServerRole	47
4.2.250 HadoopKMServerHDFSConfiguration	47
4.2.251 HadoopKMServerRole	47
4.2.252 HadoopNameNodeHDFSConfiguration	47
4.2.253 HadoopNameNodeRole	47
4.2.254 HadoopNFSGatewayHDFSConfiguration	47
4.2.255 HadoopNFSGatewayRole	47
4.2.256 HadoopPigHDFSConfiguration	47
4.2.257 HadoopPigRole	47
4.2.258 HadoopSecondaryNameNodeHDFSConfiguration	47
4.2.259 HadoopSecondaryNameNodeRole	47
4.2.260 HadoopSparkMasterHDFSConfiguration	47
4.2.261 HadoopSparkMasterRole	47
4.2.262 HadoopSparkWorkerHDFSConfiguration	47
4.2.263 HadoopSparkWorkerRole	47
4.2.264 HadoopSparkYARNHDFSConfiguration	47
4.2.265 HadoopSparkYARNRole	47
4.2.266 HadoopSqoopHDFSConfiguration	47
4.2.267 HadoopSqoopRole	47
4.2.268 HadoopStormNimbusHDFSConfiguration	47
4.2.269 HadoopStormNimbusRole	47
4.2.270 HadoopStormSupervisorHDFSConfiguration	47
4.2.271 HadoopStormSupervisorRole	47
4.2.272 HadoopTaskTrackerHDFSConfiguration	47
4.2.273 HadoopTaskTrackerRole	47
4.2.274 HadoopYARNClientHDFSConfiguration	47
4.2.275 HadoopYARNClientRole	47
4.2.276 HadoopYARNServerHDFSConfiguration	47
4.2.277 HadoopYARNServerRole	47
4.2.278 HadoopZeppelinHDFSConfiguration	48
4.2.279 HadoopZeppelinRole	48
4.2.280 HadoopZooKeeperHDFSConfiguration	48
4.2.281 HadoopZooKeeperRole	48
4.2.282 HAProxyEntry	48
4.2.283 HAProxyEntryBind	48
4.2.284 HAProxyRole	48
4.2.285 HAProxyServer	48
4.2.286 HAProxySharedSettings	48

4.2.287 IBSSwitch	48
4.2.288 IPCPerm	48
4.2.289 IPResource	48
4.2.290 Job	48
4.2.291 JobInfo	48
4.2.292 JobInfoStatistics	48
4.2.293 JobQueue	48
4.2.294 JobQueuePlaceholder	48
4.2.295 JobQueueStat	48
4.2.296 JupyterHubRole	48
4.2.297 KeepalivedEntry	48
4.2.298 KeepalivedRole	48
4.2.299 KernelModule	48
4.2.300 KeyValuePair	48
4.2.301 KibanaRole	48
4.2.302 KubeAddon	48
4.2.303 KubeAddonEnvironment	48
4.2.304 KubeCluster	48
4.2.305 KubePodInfo	48
4.2.306 KubernetesApiServerProxyRole	48
4.2.307 KubernetesApiServerRole	48
4.2.308 KubernetesControllerRole	48
4.2.309 KubernetesNodeRole	48
4.2.310 KubernetesProxyRole	48
4.2.311 KubernetesSchedulerRole	48
4.2.312 KubeRoleBinding	48
4.2.313 LabeledEntity	48
4.2.314 LicenseInfo	48
4.2.315 LiteMonitoredEntity	48
4.2.316 LiteMonitoringMeasurable	48
4.2.317 LiteNode	48
4.2.318 LoginRole	48
4.2.319 LogstashForwarderRole	48
4.2.320 LogstashServerCustomFilter	48
4.2.321 LogstashServerCustomListener	48
4.2.322 LogstashServerCustomOutput	48
4.2.323 LogstashServerElasticOutput	48
4.2.324 LogstashServerFilter	48
4.2.325 LogstashServerListener	49
4.2.326 LogstashServerLocalFileListener	49
4.2.327 LogstashServerLumberjackListener	49
4.2.328 LogstashServerOutput	49
4.2.329 LogstashServerRole	49
4.2.330 LogstashServerRSyslogFilter	49
4.2.331 LogstashServerRSyslogListener	49
4.2.332 LogstashServerStdOutput	49

4.2.333 LSFBaseJob	49
4.2.334 LSFBaseJobQueue	49
4.2.335 LSFBaseJobQueueStat	49
4.2.336 LSFGroupsSettings	49
4.2.337 LSFClientRole	49
4.2.338 LSFJob	49
4.2.339 LSFJobQueue	49
4.2.340 LSFJobQueueStat	49
4.2.341 LSFServerRole	49
4.2.342 LustreAlert	49
4.2.343 LustreClientMount	49
4.2.344 LustreFileSystem	49
4.2.345 LustreFileSystemTarget	49
4.2.346 LustreLog	49
4.2.347 LustreOverview	49
4.2.348 LustreServer	49
4.2.349 LustreServerProfile	49
4.2.350 LustreSettings	49
4.2.351 LustreTargetMap	49
4.2.352 LustreUser	49
4.2.353 LustreVolume	49
4.2.354 LustreVolumeNode	49
4.2.355 MarathonRole	49
4.2.356 MasterNode	49
4.2.357 MasterRole	49
4.2.358 MemcachedRole	49
4.2.359 MemoryInfo	49
4.2.360 MesosCluster	49
4.2.361 MesosDNSRole	49
4.2.362 MesosMasterRole	49
4.2.363 MesosProxyRole	49
4.2.364 MesosResourceUsage	49
4.2.365 MesosSlaveRole	49
4.2.366 MICHostRole	49
4.2.367 MICInfo	49
4.2.368 MICNode	49
4.2.369 MICNodeCategory	49
4.2.370 MICOverlay	49
4.2.371 MICSettings	49
4.2.372 MonitoringAction	50
4.2.373 MonitoringActionRunData	50
4.2.374 MonitoringCacheSubSystemInfo	50
4.2.375 MonitoringCategoryListExecutionFilter	50
4.2.376 MonitoringCompareExpression	50
4.2.377 MonitoringConsolidator	50
4.2.378 MonitoringDataCacheSubSystemInfo	50

4.2.379 MonitoringDataProducer	50
4.2.380 MonitoringDataProducerAggregateNode	50
4.2.381 MonitoringDataProducerAlertLevel	50
4.2.382 MonitoringDataProducerCGroup	50
4.2.383 MonitoringDataProducerClusterTotal	50
4.2.384 MonitoringDataProducerCMDaemonState	50
4.2.385 MonitoringDataProducerDeviceState	50
4.2.386 MonitoringDataProducerEC2SpotPrices	50
4.2.387 MonitoringDataProducerEthernetSwitch	50
4.2.388 MonitoringDataProducerFuture	50
4.2.389 MonitoringDataProducerGalera	50
4.2.390 MonitoringDataProducerGenerator	50
4.2.391 MonitoringDataProducerGPU	50
4.2.392 MonitoringDataProducerInternal	50
4.2.393 MonitoringDataProducerJob	50
4.2.394 MonitoringDataProducerJobMetadata	50
4.2.395 MonitoringDataProducerJobQueue	50
4.2.396 MonitoringDataProducerLua	50
4.2.397 MonitoringDataProducerMonitoringSystem	50
4.2.398 MonitoringDataProducerOpenStack	50
4.2.399 MonitoringDataProducerOpenStackHealth	50
4.2.400 MonitoringDataProducerPerpetual	50
4.2.401 MonitoringDataProducerPowerDistributionUnit	50
4.2.402 MonitoringDataProducerProcMemInfo	50
4.2.403 MonitoringDataProducerProcMount	50
4.2.404 MonitoringDataProducerProcNetDev	50
4.2.405 MonitoringDataProducerProcNetSnmp	50
4.2.406 MonitoringDataProducerProcPidStat	50
4.2.407 MonitoringDataProducerProcStat	50
4.2.408 MonitoringDataProducerProcVMStat	50
4.2.409 MonitoringDataProducerPrometheus	50
4.2.410 MonitoringDataProducerRackSensor	50
4.2.411 MonitoringDataProducerRecorder	50
4.2.412 MonitoringDataProducerScript	50
4.2.413 MonitoringDataProducerSingleLineHealthCheckScript	50
4.2.414 MonitoringDataProducerSingleLineMetricScript	50
4.2.415 MonitoringDataProducerSingleLineScript	50
4.2.416 MonitoringDataProducerSmart	50
4.2.417 MonitoringDataProducerSysBlockStat	50
4.2.418 MonitoringDataProducerSysInfo	50
4.2.419 MonitoringDataProducerTest	51
4.2.420 MonitoringDataProducerTrustedTool	51
4.2.421 MonitoringDataProducerUserCount	51
4.2.422 MonitoringDeviceStateSubSystemInfo	51
4.2.423 MonitoringDrainAction	51
4.2.424 MonitoringEmailAction	51

4.2.425 MonitoringEventAction	51
4.2.426 MonitoringExecutionFilter	51
4.2.427 MonitoringExecutionMultiplexer	51
4.2.428 MonitoringExpression	51
4.2.429 MonitoringGroupedExpression	51
4.2.430 MonitoringHealthOverview	51
4.2.431 MonitoringImageUpdateAction	51
4.2.432 MonitoringJobMetricSettings	51
4.2.433 MonitoringLuaExecutionFilter	51
4.2.434 MonitoringLuaExecutionMultiplexer	51
4.2.435 MonitoringMeasurable	51
4.2.436 MonitoringMeasurableEnum	51
4.2.437 MonitoringMeasurableHealthCheck	51
4.2.438 MonitoringMeasurableMetric	51
4.2.439 MonitoringNodeListExecutionFilter	51
4.2.440 MonitoringOverlayListExecutionFilter	51
4.2.441 MonitoringPlotterSubSystemInfo	51
4.2.442 MonitoringPowerAction	51
4.2.443 MonitoringPowerOffAction	51
4.2.444 MonitoringPowerOnAction	51
4.2.445 MonitoringPowerResetAction	51
4.2.446 MonitoringRebootAction	51
4.2.447 MonitoringReplicateConfiguration	51
4.2.448 MonitoringReplicateSource	51
4.2.449 MonitoringReplicateSubSystemInfo	51
4.2.450 MonitoringResourceExecutionFilter	51
4.2.451 MonitoringResourceExecutionMultiplexer	51
4.2.452 MonitoringRole	51
4.2.453 MonitoringScriptAction	51
4.2.454 MonitoringServiceAction	51
4.2.455 MonitoringServiceRestartAction	51
4.2.456 MonitoringServiceStartAction	51
4.2.457 MonitoringServiceStopAction	51
4.2.458 MonitoringServiceSubSystemInfo	51
4.2.459 MonitoringShutdownAction	51
4.2.460 MonitoringStorageSubSystemInfo	51
4.2.461 MonitoringSubSystemInfo	51
4.2.462 MonitoringTrigger	51
4.2.463 MonitoringTypeExecutionFilter	51
4.2.464 MonitoringTypeExecutionMultiplexer	51
4.2.465 MonitoringUndrainAction	51
4.2.466 MsgQueue	52
4.2.467 MyrinetSwitch	52
4.2.468 Network	52
4.2.469 NetworkAliasInterface	52
4.2.470 NetworkBmcInterface	52

4.2.471 NetworkBondInterface	52
4.2.472 NetworkBridgeInterface	52
4.2.473 NetworkInterface	52
4.2.474 NetworkNetMapInterface	52
4.2.475 NetworkPhysicalInterface	52
4.2.476 NetworkTunnelInterface	52
4.2.477 NetworkVLANInterface	52
4.2.478 NewNode	52
4.2.479 NginxRole	52
4.2.480 Node	52
4.2.481 NodeCategory	52
4.2.482 NodeGroup	52
4.2.483 NvidiaGPUSettings	52
4.2.484 OpenLavaCgroupsSettings	52
4.2.485 OpenLavaClientRole	52
4.2.486 OpenLavaJob	52
4.2.487 OpenLavaJobQueue	52
4.2.488 OpenLavaJobQueueStat	52
4.2.489 OpenLavaServerRole	52
4.2.490 OpenStack	52
4.2.491 OpenStackApiAgent	52
4.2.492 OpenStackApiDomain	52
4.2.493 OpenStackApiEndpoint	52
4.2.494 OpenStackApiEntity	52
4.2.495 OpenStackApiFlavor	52
4.2.496 OpenStackApiFloatingIP	52
4.2.497 OpenStackApiGroup	52
4.2.498 OpenStackApiHostAggregate	52
4.2.499 OpenStackApiHypervisor	52
4.2.500 OpenStackApiImage	52
4.2.501 OpenStackApiNetwork	52
4.2.502 OpenStackApiPort	52
4.2.503 OpenStackApiProject	52
4.2.504 OpenStackApiRole	52
4.2.505 OpenStackApiRoleAssignment	52
4.2.506 OpenStackApiRouter	52
4.2.507 OpenStackApiSecurityGroup	52
4.2.508 OpenStackApiServer	52
4.2.509 OpenStackApiService	52
4.2.510 OpenStackApiStack	52
4.2.511 OpenStackApiSubnet	52
4.2.512 OpenStackApiUser	52
4.2.513 OpenStackApiVolume	53
4.2.514 OpenStackApiVolumeSnapshot	53
4.2.515 OpenStackApiVolumeType	53
4.2.516 OpenStackAuthBackend	53

4.2.517 OpenStackAuthBackendHybrid	53
4.2.518 OpenStackAuthBackendLDAP	53
4.2.519 OpenStackAuthBackendLDAPGroupSettings	53
4.2.520 OpenStackAuthBackendLDAPProjectSettings	53
4.2.521 OpenStackAuthBackendLDAPRoleSettings	53
4.2.522 OpenStackAuthBackendLDAPUserSettings	53
4.2.523 OpenStackAuthBackendSQL	53
4.2.524 OpenStackBareMetalApiRole	53
4.2.525 OpenStackBareMetalConductorRole	53
4.2.526 OpenStackBareMetalDiscoverdDNSMasqRole	53
4.2.527 OpenStackBareMetalDiscoverdRole	53
4.2.528 OpenStackBlockStorage	53
4.2.529 OpenStackComputeApiEC2Role	53
4.2.530 OpenStackComputeApiMetadataRole	53
4.2.531 OpenStackComputeApiPlacementRole	53
4.2.532 OpenStackComputeApiRole	53
4.2.533 OpenStackComputeConductorRole	53
4.2.534 OpenStackComputeRole	53
4.2.535 OpenStackComputeSchedulerRole	53
4.2.536 OpenStackComputeVNCProxyRole	53
4.2.537 OpenStackDashboardRole	53
4.2.538 OpenStackDataProcessingApiRole	53
4.2.539 OpenStackDBaaSRole	53
4.2.540 OpenStackDefaultUserRole	53
4.2.541 OpenStackIdentityApiRole	53
4.2.542 OpenStackImageApiRole	53
4.2.543 OpenStackImageBackend	53
4.2.544 OpenStackImageBackendCeph	53
4.2.545 OpenStackImageBackendFS	53
4.2.546 OpenStackImageRegistryRole	53
4.2.547 OpenStackIntermediateStorage	53
4.2.548 OpenStackMessageQueueServerRole	53
4.2.549 OpenStackNetworkApiRole	53
4.2.550 OpenStackNetworkDHCPAgentRole	53
4.2.551 OpenStackNetworkL3AgentRole	53
4.2.552 OpenStackNetworkMetadataAgentRole	53
4.2.553 OpenStackNetworkOVSAgentRole	53
4.2.554 OpenStackNovaImageBackend	53
4.2.555 OpenStackNovaImageBackendCeph	53
4.2.556 OpenStackNovaImageBackendCow	53
4.2.557 OpenStackObjectAccountRole	53
4.2.558 OpenStackObjectApiRole	53
4.2.559 OpenStackObjectContainerRole	53
4.2.560 OpenStackObjectStoreRole	54
4.2.561 OpenStackOrchestrationApiRole	54
4.2.562 OpenStackOrchestrationRole	54

4.2.563 OpenStackSettings	54
4.2.564 OpenStackSettingsAdvanced	54
4.2.565 OpenStackSettingsAuthentication	54
4.2.566 OpenStackSettingsCMDaemonInteractions	54
4.2.567 OpenStackSettingsCollection	54
4.2.568 OpenStackSettingsCompute	54
4.2.569 OpenStackSettingsCredentials	54
4.2.570 OpenStackSettingsDatabase	54
4.2.571 OpenStackSettingsLogging	54
4.2.572 OpenStackSettingsNetworking	54
4.2.573 OpenStackSettingsPorts	54
4.2.574 OpenStackSettingsQuota	54
4.2.575 OpenStackSettingsUserPortal	54
4.2.576 OpenStackSettingsUsers	54
4.2.577 OpenStackStorage	54
4.2.578 OpenStackTelemetryAgentCentralRole	54
4.2.579 OpenStackTelemetryAgentComputeRole	54
4.2.580 OpenStackTelemetryAgentIpmiRole	54
4.2.581 OpenStackTelemetryAgentNotificationRole	54
4.2.582 OpenStackTelemetryAlarmEvaluatorRole	54
4.2.583 OpenStackTelemetryAlarmNotifierRole	54
4.2.584 OpenStackTelemetryApiRole	54
4.2.585 OpenStackTelemetryCollectorRole	54
4.2.586 OpenStackUserRole	54
4.2.587 OpenStackVolumeApiRole	54
4.2.588 OpenStackVolumeBackend	54
4.2.589 OpenStackVolumeBackend3PAR	54
4.2.590 OpenStackVolumeBackendCeph	54
4.2.591 OpenStackVolumeBackendDellStorageCenter	54
4.2.592 OpenStackVolumeBackendGPFS	54
4.2.593 OpenStackVolumeBackendNetApp	54
4.2.594 OpenStackVolumeBackendNFS	54
4.2.595 OpenStackVolumeBackendSolidFire	54
4.2.596 OpenStackVolumeBackupBackend	54
4.2.597 OpenStackVolumeBackupBackendCeph	54
4.2.598 OpenStackVolumeBackupRole	54
4.2.599 OpenStackVolumeRole	54
4.2.600 OpenStackVolumeSchedulerRole	54
4.2.601 OpenvSwitchRole	54
4.2.602 OsapiPortIP	54
4.2.603 OsapiSecurityGroupRule	54
4.2.604 OsapiStackResource	54
4.2.605 OsapiSubnetAllocationPool	54
4.2.606 OSCloudDisk	54
4.2.607 OSCloudEphemeralDisk	55
4.2.608 OSCloudExtension	55

4.2.609 OSCloudFlavor	55
4.2.610 OSCloudProvider	55
4.2.611 OSCloudRegion	55
4.2.612 OSCloudSettings	55
4.2.613 OSCloudSwapDisk	55
4.2.614 OSCloudVolumeDisk	55
4.2.615 OSService	55
4.2.616 OSServiceArray	55
4.2.617 OSServiceConfig	55
4.2.618 ParentJob	55
4.2.619 Partition	55
4.2.620 PBSJob	55
4.2.621 PBSJobQueue	55
4.2.622 PBSJobQueueStat	55
4.2.623 PbsProCgroupsSettings	55
4.2.624 PbsProClientRole	55
4.2.625 PbsProCommSettings	55
4.2.626 PbsProJob	55
4.2.627 PbsProJobQueue	55
4.2.628 PbsProJobQueueStat	55
4.2.629 PbsProMomSettings	55
4.2.630 PbsProServerRole	55
4.2.631 PDUPort	55
4.2.632 PhysicalNode	55
4.2.633 PowerDistributionUnit	55
4.2.634 PowerOperation	55
4.2.635 PowerStatus	55
4.2.636 Process	55
4.2.637 Processor	55
4.2.638 Profile	55
4.2.639 ProgramRunnerInput	55
4.2.640 ProgramRunnerKill	55
4.2.641 ProgramRunnerOutput	55
4.2.642 ProgramRunnerStatus	55
4.2.643 PrometheusQuery	55
4.2.644 PrometheusRecordingRule	55
4.2.645 ProvisioningNodeStatus	55
4.2.646 ProvisioningProcessorJob	55
4.2.647 ProvisioningRequestStatus	55
4.2.648 ProvisioningRole	55
4.2.649 ProvisioningStatus	55
4.2.650 Rack	55
4.2.651 RackPosition	55
4.2.652 RackSensor	55
4.2.653 RadosGatewayRole	55
4.2.654 RemoteNodeInstallerInteraction	56

4.2.655 RemoteSetupExecution	56
4.2.656 ResourcePool	56
4.2.657 ResourcePoolStatus	56
4.2.658 Role	56
4.2.659 S3BucketIntermediateStorage	56
4.2.660 ScaleAdvancedSettings	56
4.2.661 ScaleDynamicNodesProvider	56
4.2.662 ScaleEngine	56
4.2.663 ScaleGenericEngine	56
4.2.664 ScaleGenericTracker	56
4.2.665 ScaleHpcEngine	56
4.2.666 ScaleHpcQueueTracker	56
4.2.667 ScaleMesosEngine	56
4.2.668 ScaleMesosLoadTracker	56
4.2.669 ScalePendingWorkload	56
4.2.670 ScaleResourceProvider	56
4.2.671 ScaleServerRole	56
4.2.672 ScaleStaticNodesProvider	56
4.2.673 ScaleTracker	56
4.2.674 Semaphore	56
4.2.675 Sensor	56
4.2.676 Session	56
4.2.677 SGEClientRole	56
4.2.678 SGEJob	56
4.2.679 SGEJobQueue	56
4.2.680 SGEJobQueueStat	56
4.2.681 SGEParallelEnvironment	56
4.2.682 SGEServerRole	56
4.2.683 SharedMemory	56
4.2.684 SlaveNode	56
4.2.685 SlurmCgroupsSettings	56
4.2.686 SlurmClientRole	56
4.2.687 SlurmJob	56
4.2.688 SlurmJobQueue	56
4.2.689 SlurmJobQueueStat	56
4.2.690 SlurmServerRole	56
4.2.691 SoftwareImage	56
4.2.692 SoftwareImageFileSelection	56
4.2.693 SoftwareImageProxy	56
4.2.694 SoftwareImageRevisionInfo	56
4.2.695 StandaloneMonitoredEntity	56
4.2.696 StaticRoute	56
4.2.697 StorageNodePolicy	56
4.2.698 StorageRole	56
4.2.699 StringListObject	56
4.2.700 SubnetManagerRole	56

4.2.701 SubSystemInfo	57
4.2.702 Switch	57
4.2.703 SwitchPort	57
4.2.704 SysInfoCollector	57
4.2.705 Ticket	57
4.2.706 TorqueCgroupsSettings	57
4.2.707 TorqueClientRole	57
4.2.708 TorqueJob	57
4.2.709 TorqueJobQueue	57
4.2.710 TorqueJobQueueStat	57
4.2.711 TorqueServerRole	57
4.2.712 UCSAdaptorEthCompQueueProfile	57
4.2.713 UCSAdaptorEthGenProfile	57
4.2.714 UCSAdaptorEthInterruptProfile	57
4.2.715 UCSAdaptorEthOffloadProfile	57
4.2.716 UCSAdaptorEthRecvQueueProfile	57
4.2.717 UCSAdaptorEthUSNICProfile	57
4.2.718 UCSAdaptorEthWorkQueueProfile	57
4.2.719 UCSAdaptorExtEthIf	57
4.2.720 UCSAdaptorExtIpV6RssHashProfile	57
4.2.721 UCSAdaptorFcCdbWorkQueueProfile	57
4.2.722 UCSAdaptorFcErrorRecoveryProfile	57
4.2.723 UCSAdaptorFcGenProfile	57
4.2.724 UCSAdaptorFcInterruptProfile	57
4.2.725 UCSAdaptorFcPortFLogiProfile	57
4.2.726 UCSAdaptorFcPortPLogiProfile	57
4.2.727 UCSAdaptorFcPortProfile	57
4.2.728 UCSAdaptorFcRecvQueueProfile	57
4.2.729 UCSAdaptorFcWorkQueueProfile	57
4.2.730 UCSAdaptorHostEthIf	57
4.2.731 UCSAdaptorHostFcIf	57
4.2.732 UCSAdaptorIpV4RssHashProfile	57
4.2.733 UCSAdaptorIpV6RssHashProfile	57
4.2.734 UCSAdaptorPortProfiles	57
4.2.735 UCSAdaptorRssProfile	57
4.2.736 UCSBase	57
4.2.737 UCSBiosBootDev	57
4.2.738 UCSBiosBootDevGrp	57
4.2.739 UCSBiosSettings	57
4.2.740 UCSBiosVfAdjacentCacheLinePrefetch	57
4.2.741 UCSBiosVfAltitude	57
4.2.742 UCSBiosVfASPMSupport	57
4.2.743 UCSBiosVfConsoleRedirection	57
4.2.744 UCSBiosVfCoreMultiProcessing	57
4.2.745 UCSBiosVfCPUEnergyPerformance	57
4.2.746 UCSBiosVfCPUFrequencyFloor	57

4.2.747 UCSBiosVfCPUPerformance	57
4.2.748 UCSBiosVfCPUPowerManagement	58
4.2.749 UCSBiosVfDCUPrefetch	58
4.2.750 UCSBiosVfDemandScrub	58
4.2.751 UCSBiosVfDirectCacheAccess	58
4.2.752 UCSBiosVfDRAMClockThrottling	58
4.2.753 UCSBiosVfDramRefreshRate	58
4.2.754 UCSBiosVfEnhancedIntelSpeedStepTech	58
4.2.755 UCSBiosVfExecuteDisableBit	58
4.2.756 UCSBiosVfFRB2Enable	58
4.2.757 UCSBiosVfHardwarePrefetch	58
4.2.758 UCSBiosVfIntelHyperThreadingTech	58
4.2.759 UCSBiosVfIntelTurboBoostTech	58
4.2.760 UCSBiosVfIntelVirtualizationTechnology	58
4.2.761 UCSBiosVfIntelVTForDirectedIO	58
4.2.762 UCSBiosVfLegacyUSBSupport	58
4.2.763 UCSBiosVfLOMPortOptionROM	58
4.2.764 UCSBiosVfLvDIMMSupport	58
4.2.765 UCSBiosVfMemoryInterleave	58
4.2.766 UCSBiosVfMemoryMappedIOAbove4GB	58
4.2.767 UCSBiosVfNUMAOptimized	58
4.2.768 UCSBiosVfOnboardStorage	58
4.2.769 UCSBiosVfOnboardStorageSWStack	58
4.2.770 UCSBiosVfOSBootWatchdogTimer	58
4.2.771 UCSBiosVfOSBootWatchdogTimerPolicy	58
4.2.772 UCSBiosVfOSBootWatchdogTimerTimeout	58
4.2.773 UCSBiosVfPatrolScrub	58
4.2.774 UCSBiosVfPCIOptionROMs	58
4.2.775 UCSBiosVfPCISlotOptionROMEnable	58
4.2.776 UCSBiosVfProcessorC1E	58
4.2.777 UCSBiosVfProcessorC6Report	58
4.2.778 UCSBiosVfPStateCoordType	58
4.2.779 UCSBiosVfQPICongfig	58
4.2.780 UCSBiosVfSelectMemoryRASConfiguration	58
4.2.781 UCSBiosVfTPMSupport	58
4.2.782 UCSBiosVfUCSMBootOrderRuleControl	58
4.2.783 UCSBiosVfUSBEmulation	58
4.2.784 UCSBiosVfUSBPortsConfig	58
4.2.785 UCSBiosVfVgaPriority	58
4.2.786 UCSCommNtpProvider	58
4.2.787 UCSCommSyslog	58
4.2.788 UCSCommSyslogClient	58
4.2.789 UCSEquipmentIndicatorLed	58
4.2.790 UCSEquipmentLocatorLed	58
4.2.791 UCSFaultInst	58
4.2.792 UCSFirmwareRunning	58

4.2.793 UCSInfo	58
4.2.794 UCSLogs	58
4.2.795 UCSLsbootDef	59
4.2.796 UCSLsbootEfi	59
4.2.797 UCSLsbootLan	59
4.2.798 UCSLsbootStorage	59
4.2.799 UCSLsbootVirtualMedia	59
4.2.800 UCSStatus	59
4.2.801 UGECgroupsSettings	59
4.2.802 UGEClientRole	59
4.2.803 UGEJob	59
4.2.804 UGEJobQueue	59
4.2.805 UGEJobQueueStat	59
4.2.806 UGEParallelEnvironment	59
4.2.807 UGEServerRole	59
4.2.808 User	59
4.2.809 Validation	59
4.2.810 VersionInfo	59
4.2.811 VirtualNode	59
4.2.812 VirtualNodeSettings	59
4.2.813 VirtualSMPNode	59
4.2.814 VsmpSettings	59
4.2.815 WillChange	59
4.2.816 WlmCgroupsSettings	59
4.2.817 XeonPhiSettings	59
4.2.818 ZooKeeperCluster	59
4.2.819 ZooKeeperHostRole	59
4.3 JSON Examples	59

Preface

Welcome to the *Developer Manual* for Bright Cluster Manager 8.1.

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 8.1 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 *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.

There is also a feedback form available via Bright View, via the Account icon, , following the clickpath:

Account→Help→Feedback

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 13.2 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 `bright81`, the API reference documentation for all available objects is available in a default cluster via browser access to the URL:

```
https://bright81/userportal/downloads/python
```

The preceding access is via the User Portal (section 12.7 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 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-8.1-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

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(<name>)</code>	Find the object with a given name, <i><name></i>
<code>find(<name>, <type>)</code>	Because it is possible to give a category and node the same name, sometimes the type <i><type></i> of the object needs to be specified too
<code>getAll(<type>)</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(<object>)</code>	Add a newly created object <i><object></i> to the cluster. Only after an object is added can it be used
<code>pexec(<nodes>, <command>)</code>	Execute a command <i><command></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(<command>)</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@bright81 ~]# cd /cm/local/examples/cmd/python
[root@bright81 python]# ./printall all
Partition base
+- revision .....
| name ..... base
| clusterName ..... Bright 8.1 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@bright81 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@bright81 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`, `sgs`, `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

Parameter	Slurm	PBS Pro Torque	LSF openlava	UGE OGS (SGE)
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 `executable` line

`modules` Module files will be added to job script environment

`userdefined` These lines are added into the jobscript before the `executable` 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(): 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(): 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

Monitoring Data Producers

This chapter covers how to add a new metrics and health checks scripts with `cmsh`.

Five different types of Monitoring Data Producers can be added:

- `metric`: a script which produces a single value.
- `health check`: a script which produces a `PASS`, `FAIL`, `UNKNOWN`, or `no data` value.
- `collection`: a script that produces zero or more metrics, health checks, or a combination of both.
- `perpetual` a script that is started once over the lifetime of the Bright Cluster Manager `cmd` process. The script produces zero or more metrics, health checks, or a combination of both on its own timing mechanism.
- `prometheus` one or more URLs to Prometheus metric exporters.

A monitoring data producer cannot be plotted in `cmsh` or Bright View, because it contains no data. A producer defines measurable: metrics and/or health checks. It also generates data for these measurables, which can be plotted.

2.1 Measurables

There are three types of measurable:

- `metric`: a numeric value, or `no data`.
- `health check`: `PASS/FAIL/UNKNOWN/no data`.
- `enum metric`: one of a set of user-defined string based values, or `no data`.

2.2 Measurables Classes

All measurables are grouped into classes. A class is a user-defined free string field, with `/` as delimiters. Bright View uses this class to build a tree for easy search and access.

2.3 Metric Monitoring Data Producers

A metric data producer script generates one data point.

For example, as in the following script:

Example

```
[root@bright81 ~]# cat /path/to/my/metric
#!/bin/bash
echo $((RANDOM))
# Optionally provide extra information
echo "Extra information" >&3
```

The script can be defined as a metric script via the `monitoring setup` mode of `cmsh`:

Example

```
[bright81]% monitoring setup
[bright81->monitoring->setup]% add metric my-metric
[...my-metric]% set script /path/to/my/metric
[...my-metric]% set class My/Class
[...my-metric]% set unit B
[...my-metric]% set interval 1m
[...my-metric]% commit
```

All nodes then execute the script every minute, and produce a random number.

2.4 Health Check Monitoring Data Producers

A health check data producer script generates one data point. The data point can be one of four possible values expected of it: `PASS`, `FAIL`, `UNKNOWN`, or `no data`. Other file descriptors can be used to provide extra information.

For example, as in the following script:

Example

```
[root@bright81 ~]# cat /path/to/my/health-check
#!/bin/bash
if [ $((RANDOM)) -gt 8000 ]; then
  echo "PASS"
else
  echo "FAIL"
  # Optionally provide extra information
  echo "Extra information" >&3
fi
```

The script can be defined as a health check script via the `monitoring setup` mode of `cmsh`:

Example

```
[bright81]% monitoring setup
[bright81->monitoring->setup]% add healthcheck my-health-check
[...my-check]% set script /path/to/my/health-check
[...my-check]% set class My/Class
[...my-check]% set interval 1m
[...my-check]% commit
```

All nodes then execute the script every minute, and produce data values with roughly 75% `PASS` and 25% `FAIL`.

2.5 Collection Monitoring Data Producers

A *collection data producer* script can generate multiple data points in one run. Data points can be a combination of metrics and health checks. Collection scripts are also allowed to produce no data.

A collection script has two modes: initialize mode and sample mode.

- `initialize`: defines the measurables that data values are generated for.
- `sample`: returns the data values for all the measurables defined in initialize mode.

During normal cluster operation the initialize mode is called only once, during boot. Afterwards, the script is called in sample mode at the desired interval.

The following example combines both of the metric and health check examples from earlier on. However, this time it is written as a single script, using JSON as the output format:

Example

```
[root@bright81 ~]# cat /path/to/my/collection
#!/usr/bin/python

import sys
import json
import random

def initialize():
    metric = {"metric": "my.collection.metric",
             "unit": "B",
             "class": "My/Collection"}
    check = {"check": "my.collection.check",
            "class": "My/Collection"}
    return [metric, check]

def sample():
    metric = {"metric": "my.collection.metric",
             "value": random.randint(0, 32767)}
    check = {"check": "my.collection.check",
            "info": "random with 25% failure rate",
            "value": 'PASS' if random.randint(0, 32767) > 8000 else 'FAIL'}
    return [metric, check]

def main():
    if len(sys.argv) > 1 and sys.argv[1] == "--initialize":
        data = initialize()
    else:
        data = sample()
    print json.dumps(data, indent=4)

if __name__ == '__main__':
    main()
```

The script can be defined as a collection script via the `monitoring setup mode` of `cmsh`:

Example

```
[bright81]% monitoring setup
[bright81->monitoring->setup]% add collection my-collection
[...my-collection]% set script /path/to/my/collection
[...my-collection]% set format JSON
```

```
[...my-collection]% set interval 1m
[...my-collection]% commit
```

All nodes then execute the script every minute and produce two data points upon each execution. That is, one metric and one health check per execution.

2.6 Perpetual Monitoring Data Producers

A perpetual data producer script is a special case of a collection data producer script. It is intended to be used if the script needs permanent memory storage.

Example

```
[root@bright81 ~]# cat /path/to/my/perpetual
#!/usr/bin/python

import my_sampler_module
import json
import time

# create single instance
sampler = my_sampler_module.MySampler()
# load important data into memory
sampler.load()

# Infinite loop with its own timing
delay = 0
while True:
    time.sleep(delay)
    (definitions, values, delay) = sampler.process()
    if definitions:
        # Print new measurables
        print json.dumps(definitions)
    # Print data
    print json.dumps(values)
```

The `my_sampler_module` is the part which does the important work.

Example

```
[root@bright81 ~]# cat /path/to/my/my_sampler_module.py
class MySampler:
    def __init__(self):
        self.initialized = False
        self.definitions = None

    def load(self):
        # Do time consuming work here
        metric = {"metric": "my.collection.metric",
                 "unit": "B",
                 "class": "My/Collection"}
        check = {"check": "my.collection.check",
                 "class": "My/Collection"}
        self.definitions = [metric, check]

    def process(self):
        metric = {"metric": "my.collection.metric",
```

```

        "value": random.randint(0, 32767)}
    check = {"check" : "my.collection.check",
            "value" : 'PASS' if random.randint(0, 32767) > 8000 else 'FAIL'}
    values = metric, check
    # return definitions once, afterwards they never change
    # but new definitions could be added this way
    definitions = self.definitions
    self.definitions = None
    return definitions, values, 60

```

The script can be defined as a perpetual script via the `monitoring setup` mode of `cmsh`:

Example

```

[bright81]% monitoring setup
[bright81->monitoring->setup]% add perpetual my-perpetual
[...my-perpetual]% set script /path/to/my/perpetual
[...my-perpetual]% set format JSON
[...my-perpetual]% commit

```

2.7 Prometheus Monitoring Data Producers

Prometheus is a monitoring and alerting toolkit (<https://prometheus.io>). A Prometheus monitoring data producer script parses data from a Prometheus exporter (<https://prometheus.io/docs/instrumenting/exporters/>)

The script can be defined as a Prometheus script via the `monitoring setup` mode of `cmsh`:

Example

```

[bright81]% monitoring setup
[bright81->monitoring->setup]% add prometheus my-prometheus-exporter
[...my-prometheus-exporter]% set urls http://my.prometheus.exporter:80
[...my-prometheus-exporter]% set interval 1m
[...my-prometheus-exporter]% commit

```

If multiple URLs are defined, then only the data values from the first successful HTTP GET are used.

2.8 Node Execution Filters

By default a monitoring data producer script is executed on every node. When this is not desirable, a node execution filter should be created. A node execution filter defines the nodes on which the producer script should be executed.

For example, a filter to execute the script only on cloud nodes can be configured as follows:

Example

```

[bright81]% monitoring setup use my-check
[...my-check]% nodeexecutionfilters
[...nodeexecutionfilters]% add type Cloud
[...nodeexecutionfilters*[Cloud*]]% set cloudnode yes
[...nodeexecutionfilters*[Cloud*]]% show

```

Parameter	Value
Base type	MonitoringExecutionFilter
Name	Cloud
Type	Type
Head node	no

```
Physical node      no
Cloud node        yes
Virtual node      no
Lite node         no
[...nodeexecutionfilters*[Cloud*]]% commit
```

It is also possible to filter based on the specific resources associated with a node:

Example

```
[bright81]% monitoring setup use my-IB-check
[...my-IB-check]% nodeexecutionfilters
[...nodeexecutionfilters]% add resource IB
[...nodeexecutionfilters*[IB*]]% set resources IB
[...nodeexecutionfilters*[IB*]]% commit
```

Because of high availability, a special resource, `active`, is defined for the active head node.

Example

```
[bright81]% monitoring setup use my-metric
[...my-metric]% nodeexecutionfilters
[...nodeexecutionfilters]% active
Added active resource filter
[...nodeexecutionfilters*]% commit
```

2.9 Execution Multiplexers

By default a monitoring data producer script is executed once: the node executes the script only for itself.

However, some scripts, such as BMC samplers, must be sampled from the active head node for all nodes.

In the following example a BMC script is run on each node that has the `ipmi` or `drac` resource:

Example

```
[bright81]% monitoring setup use my-ipmi-collection
[...my-ipmi-collection]% executionmultiplexers
[...executionmultiplexers]% add resource ipmi
[...executionmultiplexers*[ipmi*]]% set resources ipmi drac
[...executionmultiplexers*[ipmi*]]% set operator OR
[...executionmultiplexers*[ipmi*]]% commit
```

If an execution multiplexer `<multiplexer>` is defined, then there should also be a node execution filter `<filter>` associated with it to restrict the number of nodes on which the script runs.

This is because having the script run on many nodes for many other nodes is unlikely to be a desired configuration.

The combination of the execution filter and the multiplexer should be read as:

for every node that matches *filter*, run script, for each node that matches *multiplexer*.

A more specific example, using two of the preceding examples, with a filter based on the resource `IB`, and multiplexers based on the `IPMI/Drac` resources, the combination should be read as:

for every node that matches `IB`, run script, for each node that matches `ipmi` or `drac`.

2.10 Monitoring Resources

Every device in Bright Cluster Manager has one or more resources. These resources are automatically calculated from:

- Roles
- Hardware
- Settings

Resources for a specific node can be viewed as follows:

Example

```
[bright81]% device use node001
[bright81]% monitoringresources
CentOS7u5
Ethernet
category:default
```

It is possible to add one or more custom resources to a device:

Example

```
[bright81]% device use node001
[bright81]% add userdefinedresources MyResource
[bright81]% append userdefinedresources MyOtherResource
[bright81]% # wait ~10 seconds for the settings to propagate
[bright81]% monitoringresources
CentOS7u5
Ethernet
category:default
MyResource
MyOtherResource
```

Any of these resources can be used to filter and multiplex monitoring data producers.

If a resources changes because of a settings change, then monitoring automatically stops or starts sampling.

2.11 Collection Monitoring Data Producers With Filter And Multiplexer

If a script has an execution multiplexer set, then it needs to determine for which nodes the script runs:

Example

```
[root@bright81~]# cat /path/to/my/collection
#!/usr/bin/python

import sys
import json
import random

def initialize(entity):
    metric = {"metric": "my.collection.metric",
             "entity": entity,
             "unit": "B",
             "class": "My/Collection"}
    check = {"check": "my.collection.check",
```

```

        "entity": entity,
        "class": "My/Collection"}
    return [metric, check]

def sample(entity):
    metric = {"metric": "my.collection.metric",
             "entity": entity,
             "value": random.randint(0, 32767)}
    check = {"check" : "my.collection.check",
            "entity": entity,
            "value" : 'PASS' if random.randint(0, 32767) > 8000 else 'FAIL'}
    return [metric, check]

def main():
    try:
        # determine for which node we are sampling
        entity = os.environ['CMD_HOSTNAME']
    except:
        sys.stderr.write('Target device not specified in environment\n')
        return

    if len(sys.argv) > 1 and sys.argv[1] == "--initialize":
        data = initialize(entity)
    else:
        data = sample(entity)
    print json.dumps(data, indent=4)

if __name__ == '__main__':
    main()

```

It can be defined with a filter to run on the active head for all nodes in the GPU category:

Example

```

[bright81]% monitoring setup
[bright81->monitoring->setup]% add collection my-collection
[...my-collection]% set script /path/to/my/collection
[...my-collection]% set format JSON
[...my-collection]% set interval 1m
[...my-collection]% nodeexecutionfilters
[...nodeexecutionfilters]% active
Added active resource filter
[...nodeexecutionfilters]% exit
[...my-collection]% executionmultiplexers
[...executionmultiplexers]% add category
[...executionmultiplexers*[GPU*]% add category GPU
[...executionmultiplexers*[GPU*]% commit

```

The script is then executed on the head, once for each node in the category of GPU.

2.12 Collection Monitoring Data Producers For Standalone Entities

Sometimes monitoring data does not belong to a Bright Cluster Manager entity.

For this reason the standalone monitored entity was added in Bright Cluster Manager 8.0.

This entity can be anything with a name and custom type.

Bright Cluster Manager does nothing with this kind of entity, except allow it to store monitoring data.

Each standalone entity which needs to be monitored should be added:

Example

```
[bright81]% monitoring standalone
[bright81->monitoring->standalone]% add MSD.0
[...standalone*[MSD.0*]]% set type Lustre
[...standalone*[MSD.0*]]% commit
[...standalone*[MSD.0*]]% add MSD.1
[...standalone*[MSD.1*]]% set type Lustre
[...standalone*[MSD.1*]]% commit
```

A script can be created that produces data for all MSD entities:

Example

```
[root@bright81 ~]# cat /path/to/my/collection
#!/usr/bin/python

import sys
import json

def initialize():
    msd_0 = {"metric": "lustre.free.space",
            "entity": "MSD.0",
            "unit": "B",
            "class": "Lustre"}
    msd_1 = {"metric": "lustre.free.space",
            "entity": "MSD.1",
            "unit": "B",
            "class": "Lustre"}
    return [msd_0, msd_1]

def sample():
    msd_0 = {"metric": "lustre.free.space",
            "entity": "MSD.0",
            "value": 12345,
            "class": "Lustre"}
    msd_1 = {"metric": "lustre.free.space",
            "entity": "MSD.1",
            "value": 54321}
    return [msd_0, msd_1]

def main():
    if len(sys.argv) > 1 and sys.argv[1] == "--initialize":
        data = initialize()
    else:
        data = sample()
    print json.dumps(data, indent=4)

if __name__ == '__main__':
    main()
```

It can be defined to run on only the active head node:

Example

```
[bright81]% monitoring setup
[bright81->monitoring->setup]% add collection my-collection
[...my-collection]% set script /path/to/my/collection
[...my-collection]% set format JSON
[...my-collection]% set interval 5m
[...my-collection]% nodeexecutionfilters
[...nodeexecutionfilters]% active
Added active resource filter
[...nodeexecutionfilters]% commit
```

The script is then executed on the active head every 5 minutes and collects one data point for each MSD.

Data for a standalone script can be viewed with the same commands as for regular Bright Cluster Manager nodes.

Example

```
[bright81]% monitoring standalone
[bright81->monitoring->standalone]% use MSD.0
[...standalone*[MSD.0*]]% latestmetricdata
...
lustre.free.space          12345          3m 47s
```


3

Monitoring Actions

This chapter covers how to manage monitoring-driven actions with `cmsh`.

3.1 Actions And Triggers

A monitoring action is a script that is executed by `CMDaemon`. It runs when triggered by the monitored data.

An action by itself does nothing—it needs a trigger (section 12.4.5 of the *Administrator Manual*) to be defined to execute the action.

By default, several actions (section 12.4.4 of the *Administrator Manual*) are predefined:

- `Drain`: Drain node (node refuses new WLM jobs)
- `Event`: Send an event to users with connected client
- `ImageUpdate`: Update the image on the node
- `PowerOff`: Power off a device
- `PowerOn`: Power on a device:
- `PowerReset`: Power reset a device
- `Reboot`: Reboot a node
- `Send e-mail to administrators`: Send e-mail
- `Shutdown`: Shutdown a node
- `Undrain`: Undrain node (node accepts new WLM jobs)
- `killprocess`: `/cm/local/apps/cmd/scripts/actions/killprocess.pl`
- `remount`: `/cm/local/apps/cmd/scripts/actions/remount`
- `testaction`: `/cm/local/apps/cmd/scripts/actions/testaction`

A new action script can be created as follows:

Example

```
[bright81]% monitoring action
[bright81->monitoring->action]% add script MyScript
[...MyScript*]% set script /path/to/MyScript
[...MyScript*]% commit
```

3.2 Time Restrictions

It is possible to allow actions to only be executed at certain times, with the `allowedtime` setting.

Example

```
[bright81]% monitoring action
[bright81->monitoring->action]% add script MyScript
[...MyScript*]% set script /path/to/MyScript
[...MyScript*]% set allowedtime "9:00-17:00"
[...MyScript*]% commit
```

More complex timing restrictions are possible:

Example

```
monday-friday9:00-17:00
monday-friday00:00-09:00;17:00-00:00;saturday-sunday
november-marchmonday-saturday13:00-17:00
may-septembermonday-friday09:00-18:00;saturday-sunday13:00-17:00
```

Further examples can be seen in section 12.4.4 of the *Administrator Manual*, page 457.

3.2.1 Time Restriction Syntax In BNF Notation

The allowed values can be written as a BNF grammar:

Example

```
<start> =
    time_intervals
    | ""
<time_intervals> = <time_interval> (; <time_interval>)*
<time_interval> = <inner_time_interval>{<time_intervals>}
<inner_time_interval> =
    <day_of_week_interval>
    | <time_of_day_interval>
    | <day_of_month_interval>
    | <month_interval>
<day_of_week_interval> =
    (<day_of_week>-<day_of_week>)
    | (<day_of_week> (, <day_of_week>)*
<day_of_week> = sunday | monday | tuesday | wednesday | thursday | friday | saturday
<time_of_day_interval> = <time_of_day>-<time_of_day>
<time_of_day>= \d?\d:\d\d
<month_interval> = (<month>-<month>)
    | (<month> (, <month>)*
<month> = january | february | march | april | may | june | july | august | september
    | october | november | december
<day_of_month_interval> = (<day_of_month>-<day_of_month>)
    | (<day_of_month> (, <day_of_month>)*
<day_of_month> = \d?\d
```

3.3 CMDaemon Environment Variables

3.3.1 Standard Environment Variables Available In Action Scripts

Name	Description
CMD_ENTITY_KEY	The unique key of the entity that triggered the action.
CMD_ENTITY_NAME	The name of the entity that triggered the action.
CMD_ENTITY_TYPE	The type of entity that triggered the action.
CMD_MEASURABLE_NAME	The name of the measurable that triggered the action.
CMD_MEASURABLE_PARAMETER	The parameter of the measurable that triggered the action.
CMD_MEASURABLE_TYPE	The type of the measurable.
CMD_VALUE	The value that triggered the action.
CMD_RAW_VALUE	The raw value.
CMD_VALUE_TIME	The time on which the value was measured.
CMD_INFO_MESSAGE	Extra information sampled along with the value.
CMD_PRODUCER_NAME	The name of the monitoring data producer that samples the measurable.

...continues

...continued

Name	Description
CMD_ACTION_NAME	The name of the action that was triggered.
CMD_TRIGGER_NAME	The name of the trigger.
CMD_TRIGGER_EXPRESSION	The expression that was evaluated.
CMD_VALUE_EVAL	The result of the evaluated expression.
CMD_VALUE_COUNT	The number of times the expression evaluated to the same value.
CMD_SEVERITY	The assigned severity of the trigger.

All action scripts have the preceding standard environment variables set.

In `cmsh`, if the action object has its `node environment` parameter set to the value `yes`, then scripts running on a node are enabled with an extended environment that provides many more `CMD_*` environment variables. Otherwise they run in the standard environment.

A list of the standard or extended environment variables can be dumped by running the system command `env > /tmp/dumpfile` within an action script, such as the test example script, and triggering the script to run.

Many of the environment variables are similar to the ones used by `initialize` and `finalize` scripts (section E.3 of the *Administrator Manual*) in the `node-installer` environment.

3.3.2 Extended Environment Variables Available To Action Scripts

If the action object has its `node environment` parameter set to the value `yes`, then scripts run in an extended environment that provides many more `CMD_*` environment variables. Otherwise they run in the standard environment of section 3.3.1.

The following table shows the additionally available environment variables with some example values:

Table 3.3.2: Environment Variables For Nodes In The Extended Environment

Variable	Example Value
CMD_ACTIVE_MASTER_IP	10.141.255.254

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_ADDED_NODES	
CMD_BASE_TYPE	
CMD_BMCIP	
CMD_BMCPASSWORD	doQNeV1qksXr590
CMD_BMCUSERID	4
CMD_BMCUSERNAME	
CMD_BMC_TYPE	2
CMD_CATEGORY	default
CMD_CEPH_MDS_SOCKET	
CMD_CEPH_MGR_SOCKET	
CMD_CEPH_MON_SOCKET	
CMD_CEPH_NAME	
CMD_CEPH_OSD_ID	
CMD_CEPH_OSD_SOCKET	
CMD_CHASSIS	chassis01
CMD_CHASSIS_IP	10.141.1.1
CMD_CHASSIS_MEMBERS	
CMD_CHASSIS_PASSWORD	secr3t
CMD_CHASSIS_SLOT	1

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_CHASSIS_USERNAME	ADMIN
CMD_CHILD_TYPE	
CMD_CLUSTERNAME	Bright 8.1 Cluster
CMD_CONFIGURATION_CREATE_DIRECTORY	
CMD_CONFIGURATION_FILENAME	
CMD_CONFIGURATION_GROUP_NAME	
CMD_CONFIGURATION_MASK	
CMD_CONFIGURATION_NAME	
CMD_CONFIGURATION_USER_NAME	
CMD_CREATE_RAMDISK_TOKEN_CATS	
CMD_CREATE_RAMDISK_TOKEN_NODES	
CMD_CURRENT_NODES	
CMD_DATA	
CMD_DELLFW_FTP_PASSWORD	
CMD_DELLFW_FTP_USERNAME	
CMD_DELLFW_PATH	
CMD_DESTINATION_REVISION	
CMD_DESTINATION_VERSION	
CMD_DEVICE_HEIGHT	1

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_DEVICE_POSITION	10
CMD_DEVICE_TYPE	ComputeNode
CMD_DIRECTOR	
CMD_DIRECTOR_IP	
CMD_DOCKER_ENDPOINTS	
CMD_EDGE_SITE	
CMD_ETCD_CA	
CMD_ETCD_CAKEY	
CMD_ETCD_CLIENT_CA	
CMD_ETCD_CLIENT_CERT	
CMD_ETCD_CLIENT_KEY	
CMD_ETCD_MEMBER_CERT	
CMD_ETCD_MEMBER_KEY	
CMD_ETHERNETSWITCH	switch01:1
CMD_EXISTING_REVISION	
CMD_EXISTING_VERSION	
CMD_EXPORTS	
CMD_FAILONMISSINGBMC	
CMD_FAIL_ON_FAILED_BMCCOMMAND	YES

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_FSEXPORIS	
CMD_FSEXPORIS_<name>_ALLOWWRITE	
CMD_FSEXPORIS_<name>_HOSTS	
CMD_FSEXPORIS_<name>_PATH	
CMD_FSMOUNTS	
CMD_FSMOUNT_<name>_DEVICE	

where <name> takes these SLASH substitutions:

<name>	example value
_SLASH_cm_SLASH_shared	\$localnfssserver:/cm/shared
_SLASH_dev_SLASH_pts	none
_SLASH_dev_SLASH_shm	none
_SLASH_home	\$localnfssserver:/home
_SLASH_proc	none
_SLASH_sys	none

CMD_FSMOUNT_<name>_FILESYSTEM

where <name> takes these SLASH substitutions:

<name>	example value
_SLASH_cm_SLASH_shared	nfs
_SLASH_dev_SLASH_pts	devpts
_SLASH_dev_SLASH_shm	tmpfs
_SLASH_home	nfs
_SLASH_proc	proc
_SLASH_sys	sysfs

CMD_FSMOUNT_<name>_MOUNTPOINT

where <name> takes these SLASH substitutions:

<name>	example value
_SLASH_cm_SLASH_shared	/cm/shared
_SLASH_dev_SLASH_pts	/dev/pts
_SLASH_dev_SLASH_shm	/dev/shm
_SLASH_home	/home
_SLASH_proc	/proc
_SLASH_sys	/sys

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_FSMOUNT_<name>_OPTIONS	
where <name> takes these SLASH substitutions:	
<name>	example value
_SLASH_cm_SLASH_shared	rsize=32768, wsize=32768, hard, intr, async
_SLASH_dev_SLASH_pts	gid=5, mode=620
_SLASH_dev_SLASH_shm	defaults
_SLASH_home	rsize=32768, wsize=32768, hard, intr, async
_SLASH_proc	defaults, nosuid
_SLASH_sys	/defaults
CMD_GATEWAY	10.141.255.254
CMD_GUID	
CMD_HAPROXY_HOST	
CMD_HOSTNAME	node004
CMD_INITRD	
CMD_INITRD_KERNEL_PARAMS	
CMD_INITRD_KERNEL_VERSION	
CMD_INITRD_TMPFS_SIZE	
CMD_INSTALLMODE	AUTO
CMD_INSTANCE_ID	
CMD_INTERFACES	BOOTIF
CMD_INTERFACE_<interface>_BOND	
CMD_INTERFACE_<interface>_BRIDGE	
CMD_INTERFACE_<interface>_DHCP	
CMD_INTERFACE_<interface>_GATEWAY	

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_INTERFACE_<interface>_IP	10.141.0.5
CMD_INTERFACE_<interface>_LANCHANNEL	
CMD_INTERFACE_<interface>_MAC	00:00:00:00:00:00
CMD_INTERFACE_<interface>_MODE	
CMD_INTERFACE_<interface>_MTU	1500
CMD_INTERFACE_<interface>_NETMASK	
CMD_INTERFACE_<interface>_REVISION	
CMD_INTERFACE_<interface>_SLAVES	
CMD_INTERFACE_<interface>_SPEED	
CMD_INTERFACE_<interface>_STARTIF	ALWAYS
CMD_INTERFACE_<interface>_TYPE	NetworkPhysicalInterface
CMD_INTERFACE_<interface>_VLANID	

In the preceding `CMD_INTERFACE_*` variables, `<interface>` can take the following substitutions for the network interface:

possible values for <interface>

BOOTIF
 drac0, drac1, drac2...
 cimc0, cimc1, cimc2...
 eth0, eth1, eth1...
 ib0, ib1, ib2...
 ilo0, ilo1, ilo2...
 ipmi0, ipmi1, ipmi2...
 rf0, rf1, rf2...
 eno1, enp0s18f2, and other
 names consistent with the
 RHEL7 interface naming
 convention

CMD_IP	10.141.0.1
CMD_JOBNODELIST	
CMD_KUBERNETES_ADMIN_CERT	

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_KUBERNETES_ADMIN_CERT_KEY	
CMD_KUBERNETES_ADMIN_KUBECONFIG	
CMD_KUBERNETES_APISERVER_ENDPOINT	
CMD_KUBERNETES_CACERT	
CMD_KUBERNETES_CLIENT_CERTIFICATE	
CMD_KUBERNETES_CLIENT_KEY	
CMD_KUBERNETES_ETCD_ACTIVE	
CMD_KUBERNETES_ETCD_CLIENT_ENDPOINTS	
CMD_KUBERNETES_KUBELET_CERTIFICATE	
CMD_KUBERNETES_KUBELET_ENDPOINT	
CMD_KUBERNETES_KUBELET_KEY	
CMD_KUBE_DNS_IP	
CMD_KUBE_DOMAIN	
CMD_KUBE_INTERNAL_NETWORK_CIDR	
CMD_KUBE_POD_NETWORK_CIDR	
CMD_KUBE_SERVICE_NETWORK_CIDR	
CMD_LOGGING_CONFIG	
CMD_MAC	FA:16:3E:64:8E:1E
CMD_MODEL	

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_MODULES	
CMD_MODULE_<name>	
CMD_MOUNTS	
CMD_NAME	
CMD_NODEGROUPS	
CMD_NODEGROUP_NAME	
CMD_NODEGROUP_UID	
CMD_OPENSTACK_CINDER_PASSWORD	
CMD_OPENSTACK_CINDER_USERNAME	
CMD_OPENSTACK_ENABLED	
CMD_OPENSTACK_GLANCE_PASSWORD	
CMD_OPENSTACK_GLANCE_USERNAME	
CMD_OPENSTACK_KEYSTONE_INTERNAL_URL	
CMD_OPENSTACK_KEYSTONE_PASSWORD	
CMD_OPENSTACK_KEYSTONE_PUBLIC_URL	
CMD_OPENSTACK_KEYSTONE_URL	
CMD_OPENSTACK_KEYSTONE_USERNAME	
CMD_OPENSTACK_MANAGER_PASSWORD	
CMD_OPENSTACK_MANAGER_USERNAME	

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_OPENSTACK_MESSAGE_QUEUE_HOSTS	
CMD_OPENSTACK_MESSAGE_QUEUE_PASSWORD	
CMD_OPENSTACK_MESSAGE_QUEUE_USERNAME	
CMD_OPENSTACK_NOVA_PASSWORD	
CMD_OPENSTACK_NOVA_USERNAME	
CMD_OPENSTACK_REGIONS	
CMD_OPENSTACK_SERVICE_TENANT	
CMD_OPENSTACK_SSL_ENABLED	
CMD_OPENSTACK_TENANTS	
CMD_OWNED_INDEX	
CMD_PARTITION	base
CMD_PASSIVE_MASTER_IP	10.141.255.253
CMD_PDUS	
CMD_PORT	8081
CMD_PORTS	
CMD_POWER_CONTROL	custom
CMD_PROTOCOL	https
CMD_RACADM_PATH	
CMD_RACK	rack01

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_RACK_HEIGHT	42
CMD_RACK_ROOM	serverroom
CMD_READ_STRING	
CMD_REMOVED_NODES	
CMD_RESOLVE_NAME	
CMD_ROLES	
CMD_SCRIPTTIMEOUT	5
CMD_SCRIPT_TIMEOUT	5
CMD_SHARED_MASTER_IP	10.141.255.252
CMD_SKIPBMC	
CMD_SOFTWAREIMAGE	default-image
CMD_SOFTWAREIMAGE_PATH	/cm/images/default-image
CMD_STATE	
CMD_STATUS	
CMD_STATUS_CLOSED	NO
CMD_STATUS_HEALTHCHECK_FAILED	NO
CMD_STATUS_HEALTHCHECK_UNKNOWN	NO
CMD_STATUS_MESSAGE	
CMD_STATUS_RESTART_REQUIRED	NO

...continues

Table 3.3.2: Environment Variables For Nodes In The Extended Environment ...continued

Variable	Example Value
CMD_STATUS_STATEFLAPPING	NO
CMD_STATUS_USERMESSAGE	
CMD_STRICTUSERID	
CMD_SUBNET_MANAGER	
CMD_SWITCH_CONTROL_SCRIPT	
CMD_SWITCH_CONTROL_SCRIPT_TIMEOUT	
CMD_SYSINFO_SYSTEM_MANUFACTURER	RDO
CMD_SYSINFO_SYSTEM_NAME	OpenStack Compute
CMD_TAG	00000000a000
CMD_TARGET_NAME	
CMD_TARGET_NODES	
CMD_TYPE	
CMD_TYPES	
CMD_UCS_DN	sys/rack-unit-1
CMD_USERDEFINED1	var1
CMD_USERDEFINED2	var2
CMD_VMLINUZ	
CMD_WRITE_STRING	

4

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/apps/cmd/etc/htdocs/userportal/download/json/index.html`.

It can also be accessed via the user portal of the cluster by clicking on the JSON API documentation link in the documentation section of the home page (Section 12.7.3 of the *Administrator Manual*).

Some examples of JSON use are given in section 4.3

4.1 Services

- 4.1.1 **auth**
- 4.1.2 **ceph**
- 4.1.3 **cert**
- 4.1.4 **cloud**
- 4.1.5 **device**
- 4.1.6 **etcd**
- 4.1.7 **gui**
- 4.1.8 **hadoop**
- 4.1.9 **job**
- 4.1.10 **keyvalue**
- 4.1.11 **kube**
- 4.1.12 **lustre**
- 4.1.13 **main**
- 4.1.14 **mesos**
- 4.1.15 **mon**
- 4.1.16 **net**
- 4.1.17 **openstack**
- 4.1.18 **part**
- 4.1.19 **proc**
- 4.1.20 **prov**
- 4.1.21 **serv**
- 4.1.22 **session**
- 4.1.23 **test**

- 4.1.24 ticket
- 4.1.25 user
- 4.1.26 zookeeper

4.2 Entities

- 4.2.1 AMDGPUSettings
- 4.2.2 AzureDataDisk
- 4.2.3 AzureDisk
- 4.2.4 AzureIntermediateStorage
- 4.2.5 AzureLocation
- 4.2.6 AzureManagedDiskParameters
- 4.2.7 AzureOSDisk
- 4.2.8 AzureProvider
- 4.2.9 AzureSettings
- 4.2.10 AzureVMSize
- 4.2.11 BadEntityManagers
- 4.2.12 BasicResource
- 4.2.13 BeeGFSAdmonRole
- 4.2.14 BeeGFSClientRole
- 4.2.15 BeeGFSManagementRole
- 4.2.16 BeeGFSMetadataRole
- 4.2.17 BeeGFSStorageRole
- 4.2.18 BigDataAdditionalTool
- 4.2.19 BigDataAdvancedSettings
- 4.2.20 BigDataCassandra
- 4.2.21 BigDataFileSystemSettings
- 4.2.22 BigDataJobManagementSettings
- 4.2.23 BigDataLoggingSettings
- 4.2.24 BigDataSecurity
- 4.2.25 BigDataSpark
- 4.2.26 BillingHistory
- 4.2.27 BMCSettings
- 4.2.28 BootRole
- 4.2.29 BurnConfig
- 4.2.30 BurnStatus
- 4.2.31 BurnTestStatus
- 4.2.32 Category
- 4.2.33 Ceph
- 4.2.34 CephMDSRole
- 4.2.35 CephMGRRole
- 4.2.36 CephMonitorRole
- 4.2.37 CephOSDBlueStoreConfig
- 4.2.38 CephOSDConfig
- 4.2.39 CephOSDFileStoreConfig
- 4.2.40 CephOSDLegacyConfig
- 4.2.41 CephOSDPool
- 4.2.42 CephOSDRole

- 4.2.43 CephState
- 4.2.44 Certificate
- 4.2.45 CertificateRequest
- 4.2.46 CertificateSubjectName
- 4.2.47 Cgroup
- 4.2.48 CgroupController
- 4.2.49 CgroupControllerBlkio
- 4.2.50 CgroupControllerCpu
- 4.2.51 CgroupControllerCpuacct
- 4.2.52 CgroupControllerCpuset
- 4.2.53 CgroupControllerDevices
- 4.2.54 CgroupControllerFreezer
- 4.2.55 CgroupControllerHugetlb
- 4.2.56 CgroupControllerMemory
- 4.2.57 CgroupControllerNetcls
- 4.2.58 CgroupControllerNetprio
- 4.2.59 CgroupControllerNs
- 4.2.60 CgroupControllerPerf
- 4.2.61 CgroupRule
- 4.2.62 CgroupSupervisorRole
- 4.2.63 Chassis
- 4.2.64 ChronosRole
- 4.2.65 ClientUserData
- 4.2.66 CloudDirectorRole
- 4.2.67 CloudGatewayRole
- 4.2.68 CloudImage
- 4.2.69 CloudJobDescription
- 4.2.70 CloudJobSubmissionStatus
- 4.2.71 CloudNode
- 4.2.72 CloudProvider
- 4.2.73 CloudRegion
- 4.2.74 CloudSettings
- 4.2.75 CloudStaticIP
- 4.2.76 CloudStorageActionData
- 4.2.77 CloudStorageNodeState
- 4.2.78 CloudType
- 4.2.79 ClusterSetup
- 4.2.80 CMDaemonBackgroundTask
- 4.2.81 CMDaemonFailover
- 4.2.82 CMDaemonFailoverGroup
- 4.2.83 CMDaemonFailoverGroupStatus
- 4.2.84 CMDaemonFailoverPeer
- 4.2.85 CMDaemonFailoverStatus
- 4.2.86 CMDaemonStatus
- 4.2.87 CMService
- 4.2.88 CMSubConfig
- 4.2.89 CMSubIntermediateStorage

- 4.2.90 ConfigFileVersion
- 4.2.91 ConfigSum
- 4.2.92 ConfigurationOverlay
- 4.2.93 Consolidator
- 4.2.94 ContainerdHostRole
- 4.2.95 ContainerInfo
- 4.2.96 CustomizationEntry
- 4.2.97 CustomizationFile
- 4.2.98 DellClustat
- 4.2.99 DellClustatGroup
- 4.2.100 DellClustatNode
- 4.2.101 DellDiskGroupInfo
- 4.2.102 DellPhysicalDiskDriveInfo
- 4.2.103 DellRAIDControllerInfo
- 4.2.104 DellSettings
- 4.2.105 DellSettingsFirmware
- 4.2.106 DellSettingsNicDevice
- 4.2.107 DellStorageInfo
- 4.2.108 DellVirtualDiskInfo
- 4.2.109 Device
- 4.2.110 DevStatus
- 4.2.111 DIGITSRole
- 4.2.112 DiskAssertion
- 4.2.113 DiskDevice
- 4.2.114 DiskInfo
- 4.2.115 DiskPartition
- 4.2.116 DiskRaid
- 4.2.117 DiskSetup
- 4.2.118 DiskVolume
- 4.2.119 DiskVolumeGroup
- 4.2.120 DockerHostRole
- 4.2.121 DockerRegistryFilesystemStorageDriver
- 4.2.122 DockerRegistryInmemoryStorageDriver
- 4.2.123 DockerRegistryRole
- 4.2.124 DockerRegistryStorageDriver
- 4.2.125 DockerStorageAufsBackend
- 4.2.126 DockerStorageBackend
- 4.2.127 DockerStorageDeviceMapperBackend
- 4.2.128 DockerStorageOverlay2Backend
- 4.2.129 DrainAction
- 4.2.130 DrainResult
- 4.2.131 EC2AMI
- 4.2.132 EC2AvailabilityZone
- 4.2.133 EC2EBSStorage
- 4.2.134 EC2EphemeralStorage
- 4.2.135 EC2Provider
- 4.2.136 EC2Region

- 4.2.137 EC2RegionAMI
- 4.2.138 EC2Settings
- 4.2.139 EC2StaticIP
- 4.2.140 EC2Storage
- 4.2.141 EC2Type
- 4.2.142 EC2VPC
- 4.2.143 ElasticSearchRole
- 4.2.144 EntityManagersMD5
- 4.2.145 EtcCluster
- 4.2.146 EtcHostRole
- 4.2.147 EthernetSwitch
- 4.2.148 FailoverRole
- 4.2.149 FakeJob
- 4.2.150 FakeJobQueue
- 4.2.151 FakeJobQueueStat
- 4.2.152 FakeWlmClientRole
- 4.2.153 FakeWlmServerRole
- 4.2.154 FileInfo
- 4.2.155 FileSyncConfig
- 4.2.156 FileSyncStatus
- 4.2.157 FlannelConfigurationRole
- 4.2.158 FlannelHostRole
- 4.2.159 FlannelNetworkingBackend
- 4.2.160 FlannelNetworkingUdpBackend
- 4.2.161 FlannelNetworkingVxLanBackend
- 4.2.162 FSExport
- 4.2.163 FSMount
- 4.2.164 FSPart
- 4.2.165 FSPartAssociation
- 4.2.166 FSPartBasicAssociation
- 4.2.167 FSPartProviderAssociation
- 4.2.168 GaleraRole
- 4.2.169 GenericDevice
- 4.2.170 GenericResource
- 4.2.171 GenericRole
- 4.2.172 GenericRoleConfiguration
- 4.2.173 GenericRoleEnvironment
- 4.2.174 GenericRoleGeneratedConfiguration
- 4.2.175 GenericRoleStaticConfiguration
- 4.2.176 GenericRoleSymlinkConfiguration
- 4.2.177 GenericRoleTemplatedConfiguration
- 4.2.178 GPUInfo
- 4.2.179 GPUSettings
- 4.2.180 GpuUnit
- 4.2.181 GPUUnitInfo
- 4.2.182 GridEngineJob
- 4.2.183 GridEngineJobQueue

4.2.184 GridEngineJobQueueStat
4.2.185 GridEngineParallelEnvironment
4.2.186 Group
4.2.187 GuiCephOsdPoolInfo
4.2.188 GuiCephOverview
4.2.189 GuiCephPgslInfo
4.2.190 GuiClusterOverview
4.2.191 GuiCompleteOpenStackOverview
4.2.192 GuiDiskUsage
4.2.193 GuiGpuUnitOverview
4.2.194 GuiHadoopHDFSDetailHBase
4.2.195 GuiHadoopHDFSDetailHDFS
4.2.196 GuiHadoopHDFSDetailMapreduce
4.2.197 GuiHadoopHDFSDetailSpark
4.2.198 GuiHadoopHDFSDetailYarn
4.2.199 GuiHadoopHDFSDetailZooKeeper
4.2.200 GuiHadoopHDFSOverview
4.2.201 GuiJob
4.2.202 GuiKubeClusterOverview
4.2.203 GuiNetworkInterface
4.2.204 GuiNodeOverview
4.2.205 GuiNodeStatus
4.2.206 GuiOpenStackOverview
4.2.207 GuiOpenStackProjectOverview
4.2.208 GuiOpenStackTenantOverview
4.2.209 GuiPDUBank
4.2.210 GuiPDUOutlet
4.2.211 GuiPDUOverview
4.2.212 GuiSwitchOverview
4.2.213 GuiSwitchPort
4.2.214 GuiWorkload
4.2.215 HadoopAccumuloMasterHDFSConfiguration
4.2.216 HadoopAccumuloMasterRole
4.2.217 HadoopAccumuloTabletHDFSConfiguration
4.2.218 HadoopAccumuloTabletRole
4.2.219 HadoopAlluxioMasterHDFSConfiguration
4.2.220 HadoopAlluxioMasterRole
4.2.221 HadoopAlluxioWorkerHDFSConfiguration
4.2.222 HadoopAlluxioWorkerRole
4.2.223 HadoopBaseConfiguration
4.2.224 HadoopCassandraHDFSConfiguration
4.2.225 HadoopCassandraRole
4.2.226 HadoopDataNodeHDFSConfiguration
4.2.227 HadoopDataNodeRole
4.2.228 HadoopDrillHDFSConfiguration
4.2.229 HadoopDrillRole
4.2.230 HadoopFlinkJobManagerHDFSConfiguration

- 4.2.231 HadoopFlinkJobManagerRole
- 4.2.232 HadoopFlinkTaskManagerHDFSConfiguration
- 4.2.233 HadoopFlinkTaskManagerRole
- 4.2.234 HadoopHBaseClientHDFSConfiguration
- 4.2.235 HadoopHBaseClientRole
- 4.2.236 HadoopHBaseServerHDFSConfiguration
- 4.2.237 HadoopHBaseServerRole
- 4.2.238 HadoopHDFS
- 4.2.239 HadoopHiveHDFSConfiguration
- 4.2.240 HadoopHiveRole
- 4.2.241 HadoopJob
- 4.2.242 HadoopJobQueue
- 4.2.243 HadoopJobQueueStat
- 4.2.244 HadoopJobTrackerHDFSConfiguration
- 4.2.245 HadoopJobTrackerRole
- 4.2.246 HadoopJournalHDFSConfiguration
- 4.2.247 HadoopJournalRole
- 4.2.248 HadoopKafkaServerHDFSConfiguration
- 4.2.249 HadoopKafkaServerRole
- 4.2.250 HadoopKMServerHDFSConfiguration
- 4.2.251 HadoopKMServerRole
- 4.2.252 HadoopNameNodeHDFSConfiguration
- 4.2.253 HadoopNameNodeRole
- 4.2.254 HadoopNFSGatewayHDFSConfiguration
- 4.2.255 HadoopNFSGatewayRole
- 4.2.256 HadoopPigHDFSConfiguration
- 4.2.257 HadoopPigRole
- 4.2.258 HadoopSecondaryNameNodeHDFSConfiguration
- 4.2.259 HadoopSecondaryNameNodeRole
- 4.2.260 HadoopSparkMasterHDFSConfiguration
- 4.2.261 HadoopSparkMasterRole
- 4.2.262 HadoopSparkWorkerHDFSConfiguration
- 4.2.263 HadoopSparkWorkerRole
- 4.2.264 HadoopSparkYARNHDFSConfiguration
- 4.2.265 HadoopSparkYARNRole
- 4.2.266 HadoopSqoopHDFSConfiguration
- 4.2.267 HadoopSqoopRole
- 4.2.268 HadoopStormNimbusHDFSConfiguration
- 4.2.269 HadoopStormNimbusRole
- 4.2.270 HadoopStormSupervisorHDFSConfiguration
- 4.2.271 HadoopStormSupervisorRole
- 4.2.272 HadoopTaskTrackerHDFSConfiguration
- 4.2.273 HadoopTaskTrackerRole
- 4.2.274 HadoopYARNClientHDFSConfiguration
- 4.2.275 HadoopYARNClientRole
- 4.2.276 HadoopYARNServerHDFSConfiguration
- 4.2.277 HadoopYARNServerRole

4.2.278 HadoopZeppelinHDFSConfiguration
4.2.279 HadoopZeppelinRole
4.2.280 HadoopZooKeeperHDFSConfiguration
4.2.281 HadoopZooKeeperRole
4.2.282 HAProxyEntry
4.2.283 HAProxyEntryBind
4.2.284 HAProxyRole
4.2.285 HAProxyServer
4.2.286 HAProxySharedSettings
4.2.287 IBSwitch
4.2.288 IPCPerm
4.2.289 IPResource
4.2.290 Job
4.2.291 JobInfo
4.2.292 JobInfoStatistics
4.2.293 JobQueue
4.2.294 JobQueuePlaceholder
4.2.295 JobQueueStat
4.2.296 JupyterHubRole
4.2.297 KeepalivedEntry
4.2.298 KeepalivedRole
4.2.299 KernelModule
4.2.300 KeyValuePair
4.2.301 KibanaRole
4.2.302 KubeAddon
4.2.303 KubeAddonEnvironment
4.2.304 KubeCluster
4.2.305 KubePodInfo
4.2.306 KubernetesApiServerProxyRole
4.2.307 KubernetesApiServerRole
4.2.308 KubernetesControllerRole
4.2.309 KubernetesNodeRole
4.2.310 KubernetesProxyRole
4.2.311 KubernetesSchedulerRole
4.2.312 KubeRoleBinding
4.2.313 LabeledEntity
4.2.314 LicenseInfo
4.2.315 LiteMonitoredEntity
4.2.316 LiteMonitoringMeasurable
4.2.317 LiteNode
4.2.318 LoginRole
4.2.319 LogstashForwarderRole
4.2.320 LogstashServerCustomFilter
4.2.321 LogstashServerCustomListener
4.2.322 LogstashServerCustomOutput
4.2.323 LogstashServerElasticOutput
4.2.324 LogstashServerFilter

4.2.325 LogstashServerListener
4.2.326 LogstashServerLocalFileListener
4.2.327 LogstashServerLumberjackListener
4.2.328 LogstashServerOutput
4.2.329 LogstashServerRole
4.2.330 LogstashServerRSyslogFilter
4.2.331 LogstashServerRSyslogListener
4.2.332 LogstashServerStdOutput
4.2.333 LSFBaseJob
4.2.334 LSFBaseJobQueue
4.2.335 LSFBaseJobQueueStat
4.2.336 LSFCgroupsSettings
4.2.337 LSFClientRole
4.2.338 LSFJob
4.2.339 LSFJobQueue
4.2.340 LSFJobQueueStat
4.2.341 LSFServerRole
4.2.342 LustreAlert
4.2.343 LustreClientMount
4.2.344 LustreFileSystem
4.2.345 LustreFileSystemTarget
4.2.346 LustreLog
4.2.347 LustreOverview
4.2.348 LustreServer
4.2.349 LustreServerProfile
4.2.350 LustreSettings
4.2.351 LustreTargetMap
4.2.352 LustreUser
4.2.353 LustreVolume
4.2.354 LustreVolumeNode
4.2.355 MarathonRole
4.2.356 MasterNode
4.2.357 MasterRole
4.2.358 MemcachedRole
4.2.359 MemoryInfo
4.2.360 MesosCluster
4.2.361 MesosDNSRole
4.2.362 MesosMasterRole
4.2.363 MesosProxyRole
4.2.364 MesosResourceUsage
4.2.365 MesosSlaveRole
4.2.366 MICHostRole
4.2.367 MICInfo
4.2.368 MICNode
4.2.369 MICNodeCategory
4.2.370 MICOverlay
4.2.371 MICSettings

4.2.372 **MonitoringAction**
4.2.373 **MonitoringActionRunData**
4.2.374 **MonitoringCacheSubSystemInfo**
4.2.375 **MonitoringCategoryListExecutionFilter**
4.2.376 **MonitoringCompareExpression**
4.2.377 **MonitoringConsolidator**
4.2.378 **MonitoringDataCacheSubSystemInfo**
4.2.379 **MonitoringDataProducer**
4.2.380 **MonitoringDataProducerAggregateNode**
4.2.381 **MonitoringDataProducerAlertLevel**
4.2.382 **MonitoringDataProducerCGroup**
4.2.383 **MonitoringDataProducerClusterTotal**
4.2.384 **MonitoringDataProducerCMDaemonState**
4.2.385 **MonitoringDataProducerDeviceState**
4.2.386 **MonitoringDataProducerEC2SpotPrices**
4.2.387 **MonitoringDataProducerEthernetSwitch**
4.2.388 **MonitoringDataProducerFuture**
4.2.389 **MonitoringDataProducerGalera**
4.2.390 **MonitoringDataProducerGenerator**
4.2.391 **MonitoringDataProducerGPU**
4.2.392 **MonitoringDataProducerInternal**
4.2.393 **MonitoringDataProducerJob**
4.2.394 **MonitoringDataProducerJobMetadata**
4.2.395 **MonitoringDataProducerJobQueue**
4.2.396 **MonitoringDataProducerLua**
4.2.397 **MonitoringDataProducerMonitoringSystem**
4.2.398 **MonitoringDataProducerOpenStack**
4.2.399 **MonitoringDataProducerOpenStackHealth**
4.2.400 **MonitoringDataProducerPerpetual**
4.2.401 **MonitoringDataProducerPowerDistributionUnit**
4.2.402 **MonitoringDataProducerProcMemInfo**
4.2.403 **MonitoringDataProducerProcMount**
4.2.404 **MonitoringDataProducerProcNetDev**
4.2.405 **MonitoringDataProducerProcNetSnmp**
4.2.406 **MonitoringDataProducerProcPidStat**
4.2.407 **MonitoringDataProducerProcStat**
4.2.408 **MonitoringDataProducerProcVMStat**
4.2.409 **MonitoringDataProducerPrometheus**
4.2.410 **MonitoringDataProducerRackSensor**
4.2.411 **MonitoringDataProducerRecorder**
4.2.412 **MonitoringDataProducerScript**
4.2.413 **MonitoringDataProducerSingleLineHealthCheckScript**
4.2.414 **MonitoringDataProducerSingleLineMetricScript**
4.2.415 **MonitoringDataProducerSingleLineScript**
4.2.416 **MonitoringDataProducerSmart**
4.2.417 **MonitoringDataProducerSysBlockStat**
4.2.418 **MonitoringDataProducerSysInfo**

- 4.2.419 **MonitoringDataProducerTest**
- 4.2.420 **MonitoringDataProducerTrustedTool**
- 4.2.421 **MonitoringDataProducerUserCount**
- 4.2.422 **MonitoringDeviceStateSubSystemInfo**
- 4.2.423 **MonitoringDrainAction**
- 4.2.424 **MonitoringEmailAction**
- 4.2.425 **MonitoringEventAction**
- 4.2.426 **MonitoringExecutionFilter**
- 4.2.427 **MonitoringExecutionMultiplexer**
- 4.2.428 **MonitoringExpression**
- 4.2.429 **MonitoringGroupedExpression**
- 4.2.430 **MonitoringHealthOverview**
- 4.2.431 **MonitoringImageUpdateAction**
- 4.2.432 **MonitoringJobMetricSettings**
- 4.2.433 **MonitoringLuaExecutionFilter**
- 4.2.434 **MonitoringLuaExecutionMultiplexer**
- 4.2.435 **MonitoringMeasurable**
- 4.2.436 **MonitoringMeasurableEnum**
- 4.2.437 **MonitoringMeasurableHealthCheck**
- 4.2.438 **MonitoringMeasurableMetric**
- 4.2.439 **MonitoringNodeListExecutionFilter**
- 4.2.440 **MonitoringOverlayListExecutionFilter**
- 4.2.441 **MonitoringPlotterSubSystemInfo**
- 4.2.442 **MonitoringPowerAction**
- 4.2.443 **MonitoringPowerOffAction**
- 4.2.444 **MonitoringPowerOnAction**
- 4.2.445 **MonitoringPowerResetAction**
- 4.2.446 **MonitoringRebootAction**
- 4.2.447 **MonitoringReplicateConfiguration**
- 4.2.448 **MonitoringReplicateSource**
- 4.2.449 **MonitoringReplicateSubSystemInfo**
- 4.2.450 **MonitoringResourceExecutionFilter**
- 4.2.451 **MonitoringResourceExecutionMultiplexer**
- 4.2.452 **MonitoringRole**
- 4.2.453 **MonitoringScriptAction**
- 4.2.454 **MonitoringServiceAction**
- 4.2.455 **MonitoringServiceRestartAction**
- 4.2.456 **MonitoringServiceStartAction**
- 4.2.457 **MonitoringServiceStopAction**
- 4.2.458 **MonitoringServiceSubSystemInfo**
- 4.2.459 **MonitoringShutdownAction**
- 4.2.460 **MonitoringStorageSubSystemInfo**
- 4.2.461 **MonitoringSubSystemInfo**
- 4.2.462 **MonitoringTrigger**
- 4.2.463 **MonitoringTypeExecutionFilter**
- 4.2.464 **MonitoringTypeExecutionMultiplexer**
- 4.2.465 **MonitoringUndrainAction**

4.2.466 **MsgQueue**
4.2.467 **MyrinetSwitch**
4.2.468 **Network**
4.2.469 **NetworkAliasInterface**
4.2.470 **NetworkBmcInterface**
4.2.471 **NetworkBondInterface**
4.2.472 **NetworkBridgeInterface**
4.2.473 **NetworkInterface**
4.2.474 **NetworkNetMapInterface**
4.2.475 **NetworkPhysicalInterface**
4.2.476 **NetworkTunnelInterface**
4.2.477 **NetworkVLANInterface**
4.2.478 **NewNode**
4.2.479 **NginxRole**
4.2.480 **Node**
4.2.481 **NodeCategory**
4.2.482 **NodeGroup**
4.2.483 **NvidiaGPUSettings**
4.2.484 **OpenLavaCgroupsSettings**
4.2.485 **OpenLavaClientRole**
4.2.486 **OpenLavaJob**
4.2.487 **OpenLavaJobQueue**
4.2.488 **OpenLavaJobQueueStat**
4.2.489 **OpenLavaServerRole**
4.2.490 **OpenStack**
4.2.491 **OpenStackApiAgent**
4.2.492 **OpenStackApiDomain**
4.2.493 **OpenStackApiEndpoint**
4.2.494 **OpenStackApiEntity**
4.2.495 **OpenStackApiFlavor**
4.2.496 **OpenStackApiFloatingIP**
4.2.497 **OpenStackApiGroup**
4.2.498 **OpenStackApiHostAggregate**
4.2.499 **OpenStackApiHypervisor**
4.2.500 **OpenStackApiImage**
4.2.501 **OpenStackApiNetwork**
4.2.502 **OpenStackApiPort**
4.2.503 **OpenStackApiProject**
4.2.504 **OpenStackApiRole**
4.2.505 **OpenStackApiRoleAssignment**
4.2.506 **OpenStackApiRouter**
4.2.507 **OpenStackApiSecurityGroup**
4.2.508 **OpenStackApiServer**
4.2.509 **OpenStackApiService**
4.2.510 **OpenStackApiStack**
4.2.511 **OpenStackApiSubnet**
4.2.512 **OpenStackApiUser**

- 4.2.513 **OpenStackApiVolume**
- 4.2.514 **OpenStackApiVolumeSnapshot**
- 4.2.515 **OpenStackApiVolumeType**
- 4.2.516 **OpenStackAuthBackend**
- 4.2.517 **OpenStackAuthBackendHybrid**
- 4.2.518 **OpenStackAuthBackendLDAP**
- 4.2.519 **OpenStackAuthBackendLDAPGroupSettings**
- 4.2.520 **OpenStackAuthBackendLDAPProjectSettings**
- 4.2.521 **OpenStackAuthBackendLDAPRoleSettings**
- 4.2.522 **OpenStackAuthBackendLDAPUserSettings**
- 4.2.523 **OpenStackAuthBackendSQL**
- 4.2.524 **OpenStackBareMetalApiRole**
- 4.2.525 **OpenStackBareMetalConductorRole**
- 4.2.526 **OpenStackBareMetalDiscoverdDNSMasqRole**
- 4.2.527 **OpenStackBareMetalDiscoverdRole**
- 4.2.528 **OpenStackBlockStorage**
- 4.2.529 **OpenStackComputeApiEC2Role**
- 4.2.530 **OpenStackComputeApiMetadataRole**
- 4.2.531 **OpenStackComputeApiPlacementRole**
- 4.2.532 **OpenStackComputeApiRole**
- 4.2.533 **OpenStackComputeConductorRole**
- 4.2.534 **OpenStackComputeRole**
- 4.2.535 **OpenStackComputeSchedulerRole**
- 4.2.536 **OpenStackComputeVNCProxyRole**
- 4.2.537 **OpenStackDashboardRole**
- 4.2.538 **OpenStackDataProcessingApiRole**
- 4.2.539 **OpenStackDBaaSRole**
- 4.2.540 **OpenStackDefaultUserRole**
- 4.2.541 **OpenStackIdentityApiRole**
- 4.2.542 **OpenStackImageApiRole**
- 4.2.543 **OpenStackImageBackend**
- 4.2.544 **OpenStackImageBackendCeph**
- 4.2.545 **OpenStackImageBackendFS**
- 4.2.546 **OpenStackImageRegistryRole**
- 4.2.547 **OpenStackIntermediateStorage**
- 4.2.548 **OpenStackMessageQueueServerRole**
- 4.2.549 **OpenStackNetworkApiRole**
- 4.2.550 **OpenStackNetworkDHCPAgentRole**
- 4.2.551 **OpenStackNetworkL3AgentRole**
- 4.2.552 **OpenStackNetworkMetadataAgentRole**
- 4.2.553 **OpenStackNetworkOVSAgentRole**
- 4.2.554 **OpenStackNovalImageBackend**
- 4.2.555 **OpenStackNovalImageBackendCeph**
- 4.2.556 **OpenStackNovalImageBackendCow**
- 4.2.557 **OpenStackObjectAccountRole**
- 4.2.558 **OpenStackObjectApiRole**
- 4.2.559 **OpenStackObjectContainerRole**

4.2.560 **OpenStackObjectStoreRole**
4.2.561 **OpenStackOrchestrationApiRole**
4.2.562 **OpenStackOrchestrationRole**
4.2.563 **OpenStackSettings**
4.2.564 **OpenStackSettingsAdvanced**
4.2.565 **OpenStackSettingsAuthentication**
4.2.566 **OpenStackSettingsCMDaemonInteractions**
4.2.567 **OpenStackSettingsCollection**
4.2.568 **OpenStackSettingsCompute**
4.2.569 **OpenStackSettingsCredentials**
4.2.570 **OpenStackSettingsDatabase**
4.2.571 **OpenStackSettingsLogging**
4.2.572 **OpenStackSettingsNetworking**
4.2.573 **OpenStackSettingsPorts**
4.2.574 **OpenStackSettingsQuota**
4.2.575 **OpenStackSettingsUserPortal**
4.2.576 **OpenStackSettingsUsers**
4.2.577 **OpenStackStorage**
4.2.578 **OpenStackTelemetryAgentCentralRole**
4.2.579 **OpenStackTelemetryAgentComputeRole**
4.2.580 **OpenStackTelemetryAgentIpmiRole**
4.2.581 **OpenStackTelemetryAgentNotificationRole**
4.2.582 **OpenStackTelemetryAlarmEvaluatorRole**
4.2.583 **OpenStackTelemetryAlarmNotifierRole**
4.2.584 **OpenStackTelemetryApiRole**
4.2.585 **OpenStackTelemetryCollectorRole**
4.2.586 **OpenStackUserRole**
4.2.587 **OpenStackVolumeApiRole**
4.2.588 **OpenStackVolumeBackend**
4.2.589 **OpenStackVolumeBackend3PAR**
4.2.590 **OpenStackVolumeBackendCeph**
4.2.591 **OpenStackVolumeBackendDellStorageCenter**
4.2.592 **OpenStackVolumeBackendGPFS**
4.2.593 **OpenStackVolumeBackendNetApp**
4.2.594 **OpenStackVolumeBackendNFS**
4.2.595 **OpenStackVolumeBackendSolidFire**
4.2.596 **OpenStackVolumeBackupBackend**
4.2.597 **OpenStackVolumeBackupBackendCeph**
4.2.598 **OpenStackVolumeBackupRole**
4.2.599 **OpenStackVolumeRole**
4.2.600 **OpenStackVolumeSchedulerRole**
4.2.601 **OpenvSwitchRole**
4.2.602 **OsapiPortIP**
4.2.603 **OsapiSecurityGroupRule**
4.2.604 **OsapiStackResource**
4.2.605 **OsapiSubnetAllocationPool**
4.2.606 **OSCloudDisk**

4.2.607 OSCloudEphemeralDisk
4.2.608 OSCloudExtension
4.2.609 OSCloudFlavor
4.2.610 OSCloudProvider
4.2.611 OSCloudRegion
4.2.612 OSCloudSettings
4.2.613 OSCloudSwapDisk
4.2.614 OSCloudVolumeDisk
4.2.615 OSService
4.2.616 OSServiceArray
4.2.617 OSServiceConfig
4.2.618 ParentJob
4.2.619 Partition
4.2.620 PBSJob
4.2.621 PBSJobQueue
4.2.622 PBSJobQueueStat
4.2.623 PbsProCgroupsSettings
4.2.624 PbsProClientRole
4.2.625 PbsProCommSettings
4.2.626 PbsProJob
4.2.627 PbsProJobQueue
4.2.628 PbsProJobQueueStat
4.2.629 PbsProMomSettings
4.2.630 PbsProServerRole
4.2.631 PDUPort
4.2.632 PhysicalNode
4.2.633 PowerDistributionUnit
4.2.634 PowerOperation
4.2.635 PowerStatus
4.2.636 Process
4.2.637 Processor
4.2.638 Profile
4.2.639 ProgramRunnerInput
4.2.640 ProgramRunnerKill
4.2.641 ProgramRunnerOutput
4.2.642 ProgramRunnerStatus
4.2.643 PrometheusQuery
4.2.644 PrometheusRecordingRule
4.2.645 ProvisioningNodeStatus
4.2.646 ProvisioningProcessorJob
4.2.647 ProvisioningRequestStatus
4.2.648 ProvisioningRole
4.2.649 ProvisioningStatus
4.2.650 Rack
4.2.651 RackPosition
4.2.652 RackSensor
4.2.653 RadosGatewayRole

4.2.654 RemoteNodeInstallerInteraction
4.2.655 RemoteSetupExecution
4.2.656 ResourcePool
4.2.657 ResourcePoolStatus
4.2.658 Role
4.2.659 S3BucketIntermediateStorage
4.2.660 ScaleAdvancedSettings
4.2.661 ScaleDynamicNodesProvider
4.2.662 ScaleEngine
4.2.663 ScaleGenericEngine
4.2.664 ScaleGenericTracker
4.2.665 ScaleHpcEngine
4.2.666 ScaleHpcQueueTracker
4.2.667 ScaleMesosEngine
4.2.668 ScaleMesosLoadTracker
4.2.669 ScalePendingWorkload
4.2.670 ScaleResourceProvider
4.2.671 ScaleServerRole
4.2.672 ScaleStaticNodesProvider
4.2.673 ScaleTracker
4.2.674 Semaphore
4.2.675 Sensor
4.2.676 Session
4.2.677 SGEClientRole
4.2.678 SGEJob
4.2.679 SGEJobQueue
4.2.680 SGEJobQueueStat
4.2.681 SGEParallelEnvironment
4.2.682 SGEServerRole
4.2.683 SharedMemory
4.2.684 SlaveNode
4.2.685 SlurmCgroupsSettings
4.2.686 SlurmClientRole
4.2.687 SlurmJob
4.2.688 SlurmJobQueue
4.2.689 SlurmJobQueueStat
4.2.690 SlurmServerRole
4.2.691 SoftwareImage
4.2.692 SoftwareImageFileSelection
4.2.693 SoftwareImageProxy
4.2.694 SoftwareImageRevisionInfo
4.2.695 StandaloneMonitoredEntity
4.2.696 StaticRoute
4.2.697 StorageNodePolicy
4.2.698 StorageRole
4.2.699 StringListObject
4.2.700 SubnetManagerRole

- 4.2.701 SubSystemInfo
- 4.2.702 Switch
- 4.2.703 SwitchPort
- 4.2.704 SysInfoCollector
- 4.2.705 Ticket
- 4.2.706 TorqueCgroupsSettings
- 4.2.707 TorqueClientRole
- 4.2.708 TorqueJob
- 4.2.709 TorqueJobQueue
- 4.2.710 TorqueJobQueueStat
- 4.2.711 TorqueServerRole
- 4.2.712 UCSAdaptorEthCompQueueProfile
- 4.2.713 UCSAdaptorEthGenProfile
- 4.2.714 UCSAdaptorEthInterruptProfile
- 4.2.715 UCSAdaptorEthOffloadProfile
- 4.2.716 UCSAdaptorEthRecvQueueProfile
- 4.2.717 UCSAdaptorEthUSNICProfile
- 4.2.718 UCSAdaptorEthWorkQueueProfile
- 4.2.719 UCSAdaptorExtEthIf
- 4.2.720 UCSAdaptorExtIpv6RssHashProfile
- 4.2.721 UCSAdaptorFcCdbWorkQueueProfile
- 4.2.722 UCSAdaptorFcErrorRecoveryProfile
- 4.2.723 UCSAdaptorFcGenProfile
- 4.2.724 UCSAdaptorFcInterruptProfile
- 4.2.725 UCSAdaptorFcPortFLogiProfile
- 4.2.726 UCSAdaptorFcPortPLogiProfile
- 4.2.727 UCSAdaptorFcPortProfile
- 4.2.728 UCSAdaptorFcRecvQueueProfile
- 4.2.729 UCSAdaptorFcWorkQueueProfile
- 4.2.730 UCSAdaptorHostEthIf
- 4.2.731 UCSAdaptorHostFclf
- 4.2.732 UCSAdaptorIpv4RssHashProfile
- 4.2.733 UCSAdaptorIpv6RssHashProfile
- 4.2.734 UCSAdaptorPortProfiles
- 4.2.735 UCSAdaptorRssProfile
- 4.2.736 UCSBase
- 4.2.737 UCSBiosBootDev
- 4.2.738 UCSBiosBootDevGrp
- 4.2.739 UCSBiosSettings
- 4.2.740 UCSBiosVfAdjacentCacheLinePrefetch
- 4.2.741 UCSBiosVfAltitude
- 4.2.742 UCSBiosVfASPMSupport
- 4.2.743 UCSBiosVfConsoleRedirection
- 4.2.744 UCSBiosVfCoreMultiProcessing
- 4.2.745 UCSBiosVfCPUEnergyPerformance
- 4.2.746 UCSBiosVfCPUFrequencyFloor
- 4.2.747 UCSBiosVfCPUPerformance

4.2.748 UCSBiosVfCPUPowerManagement
4.2.749 UCSBiosVfDCUPrefetch
4.2.750 UCSBiosVfDemandScrub
4.2.751 UCSBiosVfDirectCacheAccess
4.2.752 UCSBiosVfDRAMClockThrottling
4.2.753 UCSBiosVfDramRefreshRate
4.2.754 UCSBiosVfEnhancedIntelSpeedStepTech
4.2.755 UCSBiosVfExecuteDisableBit
4.2.756 UCSBiosVfFRB2Enable
4.2.757 UCSBiosVfHardwarePrefetch
4.2.758 UCSBiosVfIntelHyperThreadingTech
4.2.759 UCSBiosVfIntelTurboBoostTech
4.2.760 UCSBiosVfIntelVirtualizationTechnology
4.2.761 UCSBiosVfIntelVTForDirectedIO
4.2.762 UCSBiosVfLegacyUSBSupport
4.2.763 UCSBiosVfLOMPortOptionROM
4.2.764 UCSBiosVfLvDIMMSupport
4.2.765 UCSBiosVfMemoryInterleave
4.2.766 UCSBiosVfMemoryMappedIOAbove4GB
4.2.767 UCSBiosVfNUMAOptimized
4.2.768 UCSBiosVfOnboardStorage
4.2.769 UCSBiosVfOnboardStorageSWStack
4.2.770 UCSBiosVfOSBootWatchdogTimer
4.2.771 UCSBiosVfOSBootWatchdogTimerPolicy
4.2.772 UCSBiosVfOSBootWatchdogTimerTimeout
4.2.773 UCSBiosVfPatrolScrub
4.2.774 UCSBiosVfPCIOptionROMs
4.2.775 UCSBiosVfPCISlotOptionROMEnable
4.2.776 UCSBiosVfProcessorC1E
4.2.777 UCSBiosVfProcessorC6Report
4.2.778 UCSBiosVfPStateCoordType
4.2.779 UCSBiosVfQPIConfig
4.2.780 UCSBiosVfSelectMemoryRASConfiguration
4.2.781 UCSBiosVfTPMSupport
4.2.782 UCSBiosVfUCSMBootOrderRuleControl
4.2.783 UCSBiosVfUSBEmulation
4.2.784 UCSBiosVfUSBPortsConfig
4.2.785 UCSBiosVfVgaPriority
4.2.786 UCSCommNtpProvider
4.2.787 UCSCommSyslog
4.2.788 UCSCommSyslogClient
4.2.789 UCSEquipmentIndicatorLed
4.2.790 UCSEquipmentLocatorLed
4.2.791 UCSFaultInst
4.2.792 UCSFirmwareRunning
4.2.793 UCSInfo
4.2.794 UCSLogs

4.2.795 UCSLsbootDef
4.2.796 UCSLsbootEfi
4.2.797 UCSLsbootLan
4.2.798 UCSLsbootStorage
4.2.799 UCSLsbootVirtualMedia
4.2.800 UCSStatus
4.2.801 UGECgroupsSettings
4.2.802 UGEClientRole
4.2.803 UGEJob
4.2.804 UGEJobQueue
4.2.805 UGEJobQueueStat
4.2.806 UGEParallelEnvironment
4.2.807 UGEServerRole
4.2.808 User
4.2.809 Validation
4.2.810 VersionInfo
4.2.811 VirtualNode
4.2.812 VirtualNodeSettings
4.2.813 VirtualSMPNode
4.2.814 VsmptSettings
4.2.815 WillChange
4.2.816 WlmCgroupsSettings
4.2.817 XeonPhiSettings
4.2.818 ZooKeeperCluster
4.2.819 ZooKeeperHostRole

4.3 JSON Examples

complete.sh

```

#!/bin/bash

URL=https://localhost:8081/json/
user=root
pass=secretrootpassword

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

URL=https://localhost:8081/json/

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

curl -c curl.cookie.txt -i -k -X POST -d '{"service":"login", "username":"root", \
"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":"cmmain", "call":"getProfile"}' \
$URL
curl --cookie curl.cookie.txt -i -k -X POST -d '{"service":"cmmain", "call":\
"getSubjectName"}' $URL
```

devices.sh

```
#!/bin/bash
URL=https://localhost:8081/json/

if [ "$1" == "gzip" ]; then
    wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key --header='Accept-\
Encoding: gzip' --no-check-certificate --server-response -qO- $URL --post-data='{"service":\
"cmdevice", "call":"getDevices"}'
else
    wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key --no-check-cert\
ificate --server-response -qO- $URL --post-data='{"service":"cmdevice", "call":"getDevices"}'
fi
```

Tip: run as `./devices.sh | python -mjson.tool`.

loadone.sh

```
#!/bin/bash
URL=https://localhost:8081/json/

# 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' | grep -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 --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key \
--no-check-certificate --server-response -qO- $URL --post-data="{\"service\":\"cmdevice\", \
\"call\":\"getDevice\", \"arg\":\"$node\"}"` \
nkey=$(jsonval)
if [ -z $nkey ]; then
    echo $json
    exit 1
fi
echo "$node.uniqueKey = $nkey"

json=`wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key \
--no-check-certificate --server-response -qO- $URL --post-data="{\"service\":\"cmmon\", \
\"call\":\"getMonitoringMeasurable\", \"name\":\"LoadOne\"}"` \
mkey=$(jsonval)
echo "loadone.uniqueKey = $mkey"

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

cat <<EOF > /tmp/plot.json
{ "service" : "cmmon",
  "call" : "plot",
  "request" : { "entities" : [$nkey],
                "measurables" : [$mkey],
                "intervals" : 25,
                "rangeStart" : $((day*1000)),
                "rangeEnd" : $((now*1000))
              }
}
EOF
wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key \
--no-check-certificate -qO- https://master:8081/json --post-file=/tmp/plot.json | \
python -mjson.tool

```

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
URL=https://localhost:8081/json/
user=$USER
pass=secretpassword
wget --keep-session-cookies --save-cookies cookie.txt --no-check-certificate \
--server-response -qO- $URL --post-data='{"service":"login","username":"' $user' ", \
"password":"' $pass'"}'
```

logout.sh

```
#!/bin/bash
URL=https://localhost:8081/json/
wget --load-cookies cookie.txt --no-check-certificate --server-response -qO- $URL \
--post-data='{"service":"logout"}'
```

node001.sh

```
#!/bin/bash
source url

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

wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key \
--no-check-certificate --server-response -qO- $URL --post-data='{"service":"cmdevice", \
"call":"getDevice","arg":"' $node'"}' | python -mjson.tool
```

basic_information.sh

```
#!/bin/bash
URL=https://localhost:8081/json/
wget --certificate=$HOME/.cm/admin.pem --private-key=$HOME/.cm/admin.key \
--no-check-certificate --server-response -qO- $URL --post-data='{"service":"cmpart", \
"call":"getBasicEntityInformation"}'
```