CPS Operations Guide, Release 18.4.0 (Restricted Release)

First Published: 2018-09-14

Last Modified: 2018-09-14

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED “AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: www.cisco.com go trademarks. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2018 Cisco Systems, Inc. All rights reserved.

CONTENTS

PREFACE

Preface xiii

About This Guide xiii

Audience xiii

Additional Support xiii

Conventions (all documentation) xiv

Obtaining Documentation and Submitting a Service Request xv

Important Notes xv

PREFACE

RESTRICTED RELEASE xvii

CHAPTER 1

CPS Basic Operations 1

Starting and Stopping CPS 1

Starting VMs Using VMware GUI 1

Shutting Down the Cisco Policy Server Nodes 1

Policy Director (LB) or Policy Server (QNS) Nodes 1

OAM (pcrfclient) Nodes 2

sessionmgr Nodes 2

Restarting the Cisco Policy Server 3

Restarting Database Services 3

Restarting Policy Server Services 4

Restarting All Policy Server Services 4

Restarting All Policy Server Services on a Specific VM 4

Restarting Individual Policy Server Services on a Specific VM 4

Restarting Services Managed by Monit 5

Restarting Other Services 5

Restarting Subversion 5

Restarting Policy Builder 5
Restarting Control Center 5
Restarting Services on Policy Director (lb01 and lb02) 5
Recovering After a Power Outage 6
Recovery Control 6
Cluster State Monitoring 7
Controlled Startup 7
Switching Active and Standby Policy Directors 8
Determining the Active Policy Director 8
Switching Standby and Active Policy Directors 9
Backing Up and Restoring 9
Adding or Replacing Hardware 9
Export and Import Service Configurations 10

CHAPTER 2 **Managing CPS Disks** 13
Adding a New Disk 13
Prerequisites 13
ESX Server Configuration 13
Target VM Configuration 14
Update the collectd process to use the new file system to store KPIs 14
Mounting the Replication Set from Disk to tmpfs After Deployment 15
Scenario 1 – Mounting All Members of the Replication Set to tmpsf 15
Scenario 2 – Mounting Specific Members of the Replication Set to tmpfs 16
Manage Disks to Accommodate Increased Subscriber Load 16
Clone Sessionmgr01 VM 16
Disk Repartitioning of Sessionmgr01 VM 17
Cloning and Disk Repartitioning of Sessionmgr02 VM 20

CHAPTER 3 **Managing CPS Licenses** 21
Smart Software Licensing 21
Classic Licensing 21
Comparison between Licensing Models 22
Smart Accounts/Virtual Accounts 23
Request a Cisco Smart Account 24

Cisco Smart Software Manager 24
License Conversion 24
Enable Smart Licensing for CPS 25
Product ID Tags 27
Smart Licensing CLI Commands 27
License Usage Threshold 29
Configuration 29
Validation Steps 30

CHAPTER 4 **Managing CPS Interfaces and APIs** 31
CPS Interfaces and APIs 31
Control Center GUI Interface 31
CRD REST API 32
Grafana 34
HAProxy 35
JMX Interface 35
Logstash 36
LDAP SSSD 38
Configure Policy Builder 39
Configure Grafana 39
Mongo Database 42
OSGi Console 44
Policy Builder GUI 46
REST API 47
Rsyslog 47
Rsyslog Customization 48
SVN Interface 48
CPS 7.0 and Higher Releases 49
CPS Versions Earlier than 7.0 50
TACACS+ Interface 51
Unified API 51
Accessing the CPS CLI 52
Multi-user Policy Builder 53
Create Users 53

Revert Configuration 54
Publishing Data 55
Control Center Access 55
Add a Control Center User 55
Update Control Center Mapping 56
Multiple Concurrent User Sessions 57
Configure Session Limit 58
Configure Session Timeout 58
Important Notes 59
Enabling Authentication and Authorization for CRD API 59
Unified API Security: Access Privileges 61
Enable Authentication for Unified API 62
WSDL and Schema Documentation 63
Enabling Unified API Access on HTTP Port 8080 63
TACACS+ 66
Overview 66
TACACS+ Service Requirements 66
Caching of TACACS+ Users 67
Reading Log Files 68
CRD APIs 69
Limitations 69
Setup Requirements 69
Policy Server 69
Policy Builder 69
Architecture 74
MongoDB 74
Caching 74
API Endpoints and Examples 75
Query API 75
Create API 76
Update API 77
Delete API 77
Data Comparison API 78
Table Drop API 79

Export API 80
Import API 81
Import Single File API 82
Snapshot POST API 82
Snapshot GET API 83
Revert API 84
Tips for Usage 85
View Logs 85

CHAPTER 5 **Tracking CPS GUI and API Usage** 87
Track Usage 87
Capped Collection 87
PurgeAuditHistoryRequests 87
AuditRequests 87
Operation 88
Initial Setup 88
Read Requests 88
APIs 89
Querying 89
Purging 89
Purge History 90
Control Center 90
PurgeAuditHistoryRequest 90
QueryAuditHistoryRequest 91
Policy Builder 93
Reporting 93
Audit Configuration 94
Pre-configured auditd 98

CHAPTER 6 **Prometheus and Grafana** 101
Introduction 101
Enable Prometheus 101
Add Datasource in Grafana for Prometheus 103
Grafana 104

Additional Grafana Documentation 104
Configure Grafana Users using CLI 104
Add First User 104
Add Another User 105
Delete a User 105
Connect to Grafana 105
Grafana Administrative User 106
Log in as Grafana Admin User 106
Change Grafana Admin User Credentials 107
Add a Grafana User 107
Change the Role of Grafana User 109
Add an Organization 110
Move Grafana User to another Organization 111
Configure Grafana for First Use 112
Migrate Existing Grafana Dashboards 112
Manual Dashboard Configuration using Grafana 114
Create a New Dashboard Manually 114
Configure Data Points for the Panel 116
Configure Useful Dashboard Panels 119
Updating Imported Templates 121
Copy Dashboards and Users to pcrfclient02 121
Configure Garbage Collector KPIs 122
Backend Changes 122
Frontend Changes 123
Export and Import Dashboards 125
Export Dashboard 125
Import Dashboard 125
Export Graph Data to CSV 126
Session Consumption Report 128
Introduction 128
Data Collection 128
Logging 128
Performance 128
Log Rotation 128

Sample Report 129

CHAPTER 7 Managing High Availability in CPS 131

Porting All-In-One Policy Builder Configuration to HA 131

Prerequisites 131

Porting the Policy Builder Configuration 131

HAProxy 134

HAProxy Service Operations 134

Diagnostics 134

Service Commands 134

HAProxy Statistics 134

Changing HAProxy Log Level 135

Expanding an HA Deployment 135

Typical Scenarios When Expansion is Necessary 136

Hardware Approach to Expanding 136

High Availability Consequences 136

Adding a New Blade 136

Component (VM Node) Approach to Expanding 137

Adding Additional Component 137

Enable SSL 137

CHAPTER 8 CPS Statistics 139

Bulk Statistics Overview 139

Grafana 140

CPS Statistics 140

Overview 140

CPS Statistic Types 142

Diameter Statistics 142

LDAP Statistics 142

System Statistics 142

Engine Statistics 143

MOG API Statistics 143

Error Statistics Definitions 143

Bulk Statistics Collection 144

Retention of CSV Files 145
Configuring Logback.xml 145
Restarting the Collectd Service 145
Adding Realm Names to Diameter Statistics 145
CPS KPI Monitoring 146
System Health Monitoring KPIs 146
Session Monitoring KPIs 150
Diameter Monitoring KPIs 152
Example CPS Statistics 169
Sample CSV Files 169
Sample Output 170

CHAPTER 9 Working with CPS Utilities 171
Policy Tracing and Execution Analyzer 171
Architecture 171
Administering Policy Traces 171
Managing Trace Rules using trace_ids.sh 172
Managing Trace Results using trace.sh 173
Policy Trace Database 174
Configure Traces Database in Policy Builder 174
Network Cutter Utility 175
Policy Builder Configuration Reporter 175
CRD Generator Conversion Tool 176
Policy Builder Configuration Converter Conversion Tool 179
Modifying Audit Rule File 180

CHAPTER 10 CPS Commands 183
about.sh 184
adduser.sh 184
auditrpms.sh 185
build_all.sh 185
build_etc.sh 187
build_set.sh 187
capture_env.sh 188

change_passwd.sh 189
cleanup_license.sh 189
component_alarm_reports.py 189
copytoall.sh 190
diagnostics.sh 191
dump_utility.py 194
list_installed_features.sh 198
reinit.sh 200
restartall.sh 200
restartqns.sh 201
runonall.sh 201
service 202
session_cache_ops.sh 202
Syntax 202
Options 202
Executable on VMs 206
set_priority.sh 206
startall.sh 207
startqns.sh 208
statusall.sh 208
stopall.sh 210
stopqns.sh 211
summaryall.sh 211
sync_times.sh 224
syncconfig.sh 225
terminatesessions 225
show 227
cancel 228
top_qps.sh 228
Diameter Synchronization Message Behavior 229
vm-init.sh 230

Preface

- About This Guide, on page xiii
- Audience, on page xiii
- Additional Support, on page xiii
- Conventions (all documentation), on page xiv
- Obtaining Documentation and Submitting a Service Request, on page xv
- Important Notes, on page xv

About This Guide

This document is a part of the Cisco Policy Suite documentation set.

For information about available documentation, see the *CPS Documentation Map* for this release at Cisco.com.

Audience

This guide is best used by these readers:

- Network administrators
- Network engineers
- Network operators
- System administrators

This document assumes a general understanding of network architecture, configuration, and operations.

Additional Support

For further documentation and support:

- Contact your Cisco Systems, Inc. technical representative.
- Call the Cisco Systems, Inc. technical support number.
- Write to Cisco Systems, Inc. at support@cisco.com.

- Refer to support matrix at https://www.cisco.com/c/en/us/support/index.html and to other documents related to Cisco Policy Suite.

Conventions (all documentation)

This document uses the following conventions.

Conventions	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
courier font	Terminal sessions and information the system displays appear in courier font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means reader take note. Notes contain helpful suggestions or references to material not covered in the manual.

Caution Means reader be careful. In this situation, you might perform an action that could result in equipment damage or loss of data.

Warning

IMPORTANT SAFETY INSTRUCTIONS.

Means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Note

Regulatory: Provided for additional information and to comply with regulatory and customer requirements.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, using the Cisco Bug Search Tool (BST), submitting a service request, and gathering additional information, see What's New in Cisco Product Documentation.

To receive new and revised Cisco technical content directly to your desktop, you can subscribe to the . RSS feeds are a free service.

Important Notes

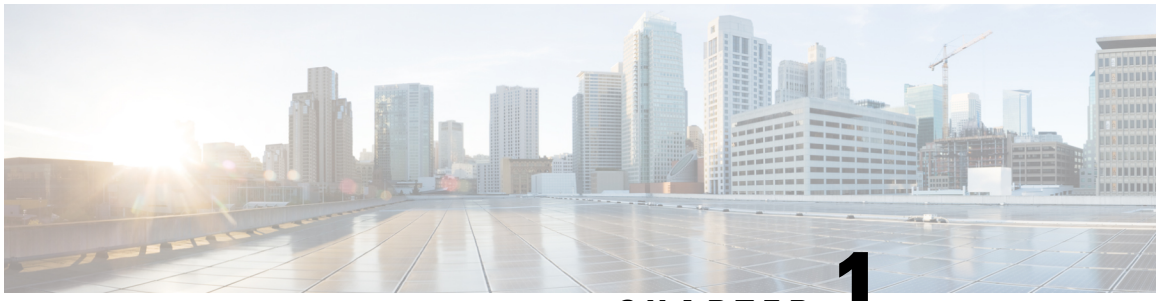
Important

Any feature or GUI functionality that is not documented may not be supported in this release or may be customer specific, and must not be used without consulting your Cisco Account representative.

RESTRICTED RELEASE

Important This is a Short Term Support (STS) release with availability and use restrictions. Contact your Cisco Account or Support representatives for more information.

RESTRICTED RELEASE



CHAPTER 1

CPS Basic Operations

- Starting and Stopping CPS, on page 1
- Restarting the Cisco Policy Server, on page 3
- Recovering After a Power Outage, on page 6
- Backing Up and Restoring, on page 9
- Adding or Replacing Hardware, on page 9
- Export and Import Service Configurations, on page 10

Starting and Stopping CPS

This section describes how to start and stop Cisco Policy Server nodes, VMs, and services.

Starting VMs Using VMware GUI

Step 1 Start a VMware vSphere session.

Step 2 Right-click the VM and select **Power** > **Power On**.

Important If the Policy Server (QNS) VM was previously powered off, it must be powered on only during Maintenance Window or low traffic time. If the VM is powered on during high traffic, then when the qns java process comes up and it immediately starts taking up load. As a result there can be timeouts and high CPU until around 60 seconds from the Policy Server (QNS) VM during the JVM hotspot warmup time. Once the JVM warmup phase is completed, the VM must be able to handle traffic smoothly.

Step 3 After the VM has started, log into the VM from Cluster Manager and verify that the processes are running.

Shutting Down the Cisco Policy Server Nodes

The following sections describe the commands to shut down the Cisco Policy Server nodes:

Policy Director (LB) or Policy Server (QNS) Nodes

Step 1 SSH to the lbxx or qnsxx node from Cluster Manager:

`ssh lbxx` or `ssh qnsxx`

Step 2 Stop all CPS processes on the node:

```
/usr/bin/monit stop all
```

Step 3 Check the status of all the processes. Verify that all processes are stopped before proceeding.

```
/usr/bin/monit summary
```

Step 4 Stop the monit process:

```
service monit stop
```

Step 5 Shut down lbxx/qnsxx:

```
shutdown -h now
```

OAM (pcrfclient) Nodes

Step 1 SSH to the pcrfclient*xx* node from Cluster Manager:

`ssh pcrfclient`*xx*

Step 2 Stop all CPS processes on the node:

```
/usr/bin/monit stop all
```

Step 3 Check the status of all the processes. Verify that all processes are stopped before proceeding:

```
/usr/bin/monit summary
```

Step 4 Stop the monit process:

```
service monit stop
```

Step 5 Stop the licenses process:

```
service lmgrd stop
```

Step 6 Shut down pcrfclient*xx*:

```
shutdown -h now
```

sessionmgr Nodes

Step 1 SSH to the sessionmgr*xx* node from Cluster Manager:

`ssh sessionmgr`*xx*

Step 2 Stop all CPS processes on the node:

```
/usr/bin/monit stop all
```

Step 3 Check the status of all the processes. Verify that all processes are stopped before proceeding:

```
/usr/bin/monit summary
```

Step 4 Stop the monit process:

```
service monit stop
```

Step 5 For CPS nodes, such as sessionMgrs, there are mongo processes running that require special steps to stop. First, determine which processes are running by executing:

```
ls /etc/init.d/sessionmgr*
```

Step 6 Make sure the mongo replica set is in secondary:

```
/usr/bin/mongo --port $PORT --eval "rs.stepDown(10)"
```

where, PORT is the port number found in the previous step, such as 27717.

Step 7 Stop the MongoDB processes.

For example:

```
systemctl stop sessionmgr-27717
```

Step 8 Shut down sessionmgr*xx*:

```
shutdown -h now
```

Restarting the Cisco Policy Server

CPS is composed of a cluster of nodes and services. This section describes how to restart the different services running on various CPS nodes.

Restarting Database Services

Each database port and configuration is defined in the `/etc/broadhop/mongoConfig.cfg` file.

The scripts that start/stop the database services can be found in the `/usr/bin` directory on the CPS nodes.

To stop and start a database, log into each Session Manager VM and execute the commands as shown below. For example, to restart the sessionmgr 27717 database, execute:

```
systemctl stop sessionmgr-27717
```

```
systemctl start sessionmgr-27717
```

or:

```
systemctl restart sessionmgr-27717
```

Note It is important not to stop and start all of the databases in the same replica-set at the same time. As a best practice, stop and start databases one at a time to avoid service interruption.

Restarting Policy Server Services

If the Policy Server (QNS) VM was previously powered off, it must be powered on only during Maintenance Window or low traffic time. If the VM is powered on during high traffic, then when the qns java process comes up and it immediately starts taking up load. As a result there can be timeouts and high CPU until around 60 seconds from the Policy Server (QNS) VM during the JVM hotspot warmup time. Once the JVM warmup phase is completed, the VM must be able to handle traffic smoothly.

Restarting All Policy Server Services

To restart all Policy Server (QNS) services on all VMs, execute the following from the Cluster Manager:

```
/var/qps/bin/control/restartall.sh
```

Note This script only restarts the Policy Server (QNS) services. It does not restart any other services.

Use summaryall.sh or statusall.sh to see details about these services.

Restarting All Policy Server Services on a Specific VM

To restart all Policy Server (QNS) services on a single CPS VM, execute the following from the Cluster Manager:

```
/var/qps/bin/control/restartqns.sh <hostname>
```

where <*hostname*> is the CPS node name of the VM (qns01, qns02, lb01, pcrfclient01, and so on).

Restarting Individual Policy Server Services on a Specific VM

Step 1 Log into the specific VM.

Step 2 To determine what Policy Server (QNS) services are currently running on the VM, execute:

```
monit summary
```

Output similar to the following appears:

```
The Monit daemon 5.5 uptime: 1d 17h 18m

Process 'qns-4' Running
Process 'qns-3' Running
Process 'qns-2' Running
Process 'qns-1' Running
```

Step 3 Execute the following commands to stop and start the individual Policy Server (QNS) process:

```
monit stop qns-<instance id>

monit start qns-<instance id>
```

Restarting Services Managed by Monit

The Monit service manages many of the services on each CPS VM.

To see a list of services managed by `monit` on a VM, log in to the specific VM and execute:

```
monit summary
```

To stop and start all services managed by `monit`, log in to the specific VM and execute the following commands:

```
monit stop all
```

```
monit start all
```

To stop and start a specific service managed by Monit, log in to the specific VM and execute the following commands:

```
monit stop <service_name>
```

```
monit start <service_name>
```

where <*service_name*> is the name as shown in the output of the `monit summary` command.

Restarting Other Services

Restarting Subversion

To restart Subversion (SVN) on OAM (pcrfclient) nodes, execute:

```
service httpd restart
```

Restarting Policy Builder

To restart Policy Builder on OAM (pcrfclient) nodes (pcrfclient01/pcrfclient02), execute:

```
monit stop qns-2
```

```
monit start qns-2
```

Restarting Control Center

To restart Control Center on OAM (pcrfclient) nodes (pcrfclient01/pcrfclient02), execute:

```
monit stop qns-1
```

```
monit start qns-1
```

Restarting Services on Policy Director (lb01 and lb02)

The following commands are used to restart the services on the Policy Director (lb) nodes only (lb01 and lb02).

Step 1 Login to lb01/lb02.

Step 2 To restart the service that controls the virtual IPs (lbvip01 and lbvip02 are virtual IP addresses shared between lb01 and lb02 for High Availability), execute the following command:

```
monit restart corosync
```

Step 3 To restart the service that balances and forwards IP traffic (port forwarding service) from lb01/lb02 to other CPS nodes, execute:

```
monit restart haproxy
```

Recovering After a Power Outage

If there is a controlled or uncontrolled power outage, the following power on procedures should be followed to bring the system up properly.

Step 1 Power ON the Cluster Manager.

Step 2 Power ON pcrfclient01.

Step 3 Power ON all Session Manager nodes (sessionmgr0x).

Step 4 Validate that the databases are all online by running:

```
diagnostics.sh --get_replica_status
```

Note If a member is shown in an unknown state, it is likely that the member is not accessible from one of other members, mostly an arbiter. In that case, you must go to that member and check its connectivity with other members.

Also, you can login to mongo on that member and check its actual status.

Step 5 Power ON Policy Director node 2 (lb02).

Step 6 Power ON Policy Director node 1 (lb01).

Step 7 Power ON all Policy Server (QNS) nodes.

Step 8 Power ON pcrfclient02.

Step 9 On pcrfclient01, run the following commands to reinitialize the services:

```
monit stop all
```

```
monit start all
```

Step 10 Run `diagnostics.sh` to validate system is functioning properly.

Recovery Control

Due to the operational inter-dependencies within the CPS, it is necessary for some CPS services and components to become active before others.

CPS can monitor the state of the cluster through the various stages of startup. It also includes functionality to allow the system to gracefully recover from unexpected outages.

Cluster State Monitoring

CPS can monitor the state of the services and components of the cluster from the OAM (pcrfclient) VMs. By default, this functionality is disabled.

This functionality can be enabled by setting the cluster_state_monitor option to true in the CPS Deployment Template (Excel spreadsheet).

To update an existing deployment to support this functionality, modify this setting in your CPS Deployment Template and redeploy the csv files as described in the *CPS Installation Guide for VMware*.

This monitoring system reports the state of the system as an integer value as described in the following table:

Table 1: Cluster State Monitoring

Cluster State	Description	Values
0	unknown state/pre-inspection state	The system will report '0' until both conditions have been met under '1': lbvip02 is UP AND databases are accessible. Various systems can be coming online while a '0' state is being reported and does not automatically indicate an error. Even if the system cannot proceed to '1' state, Policy Builder and Control Center UIs should be available in order to manage or troubleshoot the system.
1	lbvip02 is alive and all databases in `/etc/broadhop/mongoConfig.cfg` have an accessible primary	All backend databases must be available and the lbvip02 interface must be UP for the system to report this state.
2	lbvip02 port 61616 is accepting TCP connections	Backend Policy Server (QNS) processes access lbvip02 on this port. When this port is activated, it indicates that Policy Server (QNS) processes can proceed to start.
3	at least 50% of backend Policy Server (QNS) processes are alive	Once sufficient capacity is available from the backend processes, the Diameter protocol endpoint processes are allowed to start.

The current cluster state is reported in the following file on the OAM (pcrfclient):
`/var/run/broadhop.cluster_state`

The `determine_cluster_state` command logs output of the cluster state monitoring process into `/var/log/broadhop/determine_cluster_state.log`.

Controlled Startup

In addition to the monitoring functionality, CPS can also use the cluster state to regulate the startup of some of the CPS services pending the appropriate state of the cluster.

By default this functionality is disabled. It can be enabled for the entire CPS cluster, or for troubleshooting purposes can be enabled or disabled on a per-VM basis.

Note Cluster State Monitoring must be enabled for Controlled Startup to function.

Enable/Disable For All VMs in Cluster

The Controlled Startup functionality is enabled by the presence of the `/etc/broadhop/cluster_state` file.

To enable this feature on all CPS VMs in the cluster, execute the following commands on the Cluster Manager VM to create this file and to use the syncconfig.sh script to push those changes out to the other VMs.

```
touch /etc/broadhop/cluster_state
```

```
syncconfig.sh
```

To disable this feature on all VMs in the cluster, remove the cluster_state file on the Cluster Manager VM and sync the configuration:

```
rm /etc/broadhop/cluster_state
```

```
syncconfig.sh
```

Enable/Disable For Specific VM

To enable this feature on a specific VM, create a `/etc/broadhop/cluster_state` file on the VM:

`touch /etc/broadhop/cluster_state`

To disable this feature again on a specific VM, delete the `/etc/broadhop/cluster_state` file on the VM:

`rm /etc/broadhop/cluster_state`

Note This is temporary measure and should only be used for diagnostic purposes. Local modifications to a VM can be overwritten under various circumstances, such as running **syncconfig.sh**.

Switching Active and Standby Policy Directors

In CPS, the active and standby strategy applies only to the Policy Directors (lb). The following are the two Policy Directors in the system:

- lb01
- lb02

Determining the Active Policy Director

Step 1 Log in to the pcrfclient01 VM.

Step 2 Run the following command to SSH to the active Policy Director (typically lb01):

```
ssh lbvip01
```

Step 3 You can also confirm an active Policy Director by ensuring it has the virtual IP (VIP) associated with it by running the following command:

```
ifconfig -a
```

If you see the eth0:0 or eth1:0 interfaces present in the list and marked as "UP" then that is the active Policy Director.

For example:

```
eth0:0   Link encap:Ethernet  HWaddr 00:0C:29:CD:7E:4C
         inet addr:172.26.241.240  Bcast:172.26.241.255  Mask:255.255.254.0
     --> UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1 The passive or standby load balancer will
 not have active VIPs
    shown in the
    ifconfig -a output (no eth0:0 and eth1:0).
```

Switching Standby and Active Policy Directors

Step 1 Log in to the active Policy Director (lb) VM. See Determining the Active Policy Director, on page 8 for details to determine which Policy Director is active.

Step 2 Restart the Heartbeat service using the following command:

```
monit restart corosync
```

This command will force the failover of the VIP from the active Policy Director to the standby Policy Director.

Step 3 To confirm the switchover, SSH to the other Policy Director VM and run the following command to determine if the VIP is now associated with this VM:

```
ifconfig -a
```

If you see the eth0:0 or eth1:0 interfaces in the list and marked as "UP" then that is the active Policy Director.

Backing Up and Restoring

As a part of routine operations, it is important to make backups so that if there are any failures, the system can be restored. Do not store backups on system nodes.

For detailed information about backup and restore procedures, see the *CPS Backup and Restore Guide*.

Adding or Replacing Hardware

Hardware replacement is usually performed by the hardware vendor with whom your company holds a support contract.

Hardware support is not provided by Cisco. The contact persons and scheduling for replacing hardware is made by your company.

Before replacing hardware, always make a backup. See the *CPS Backup and Restore Guide*.

Unless you have a readily available backup solution, use VMware Data Recovery. This solution, provided by VMware under a separate license, is easily integrated into your CPS environment.

The templates you download from the Cisco repository are partially pre-configured but require further configuration. Your Cisco technical representative can provide you with detailed instructions.

Note You can download the VMware software and documentation from the following location:

http://www.vmware.com/

Export and Import Service Configurations

You can export and import service configurations for the migration and replication of data. You can use the export/import functions to back up both configuration and environmental data or system-specific information from the configuration for lab-to-production migration.

You can import the binary in the following two ways:

- Import the binary produced by export - All configuration exported will be removed (If environment is included, only environment will be removed. If environment is excluded, environment will not be removed). The file passed is created from the export API.
- Additive Import - Import the package created manually by adding configuration. The new configurations get added into the server without impacting the existing configurations. The import is allowed only if the CPS running version is greater than or equal to the imported package version specified in the configuration.

Step 1 In a browser, navigate to the export/import page, available at the following URLs:

HA/GR: https://<<*lbvip01*>:7443/doc/import.html

All-In-One (AIO): http://<*ip*>:7070/doc/import.html

Step 2 Enter the API credentials.

Step 3 Select the file to be imported/exported.

The following table describes the export/import options:

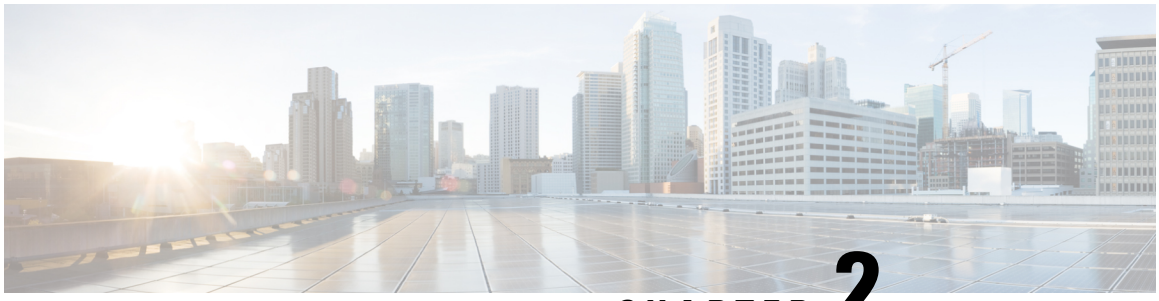
Table 2: Export and Import Options

Option	Description
Export	
All data	Exports service configuration with environment data, which acts as a complete backup of both service configurations and environmental data.

Option	Description
Exclude environment	Exports without environment data, which allows exporting configuration from a lab and into another environment without destroying the new system's environment-specific data.
Only environment	Exports only environment data, which provides a way to back up the system-specific environmental information.
Export URL	Found in Policy Builder or viewed directly in Subversion.
Export File Prefix	Provide a name (prefix) for the export file. **Note:** The exported filename automatically includes the date and time when the export was performed, for example: *prefix*_2016-01-12_11-03-56_3882276668.cps **Note:** The file extension **.cps** is used so that the file is not opened or modified by mistake by another application. The file should be used for export/import purposes only.
Import	
Import URL	URL is updated/created. We recommend importing to a new URL and use Policy Builder to verify/publish.
Commit Message	Message recorded with the import. Provide details that are useful to record.

After you select the file, the file's information is displayed.

Step 4 Select **Import** or **Export**.
CPS displays response messages that indicate the status of the export/import.



CHAPTER 2

Managing CPS Disks

- Adding a New Disk, on page 13
- Mounting the Replication Set from Disk to tmpfs After Deployment, on page 15
- Manage Disks to Accommodate Increased Subscriber Load, on page 16

Adding a New Disk

This section describes the procedures needed to add a new disk to a VM.

Prerequisites

- All the VMs were created using the deployment process.
- This procedure assumes the datastore that will be used to have the virtual disk has sufficient space to add the virtual disk.
- This procedure assumes the datastore has been mounted to the VMware ESX server, regardless of the backend NAS device (SAN or iSCSI, etc).

ESX Server Configuration

Step 1 Login to the ESX server shell, and make sure the datastore has enough space:

```
vmkfstools -c 4g /vmfs/volumes/datastore_name/VMNAME/xxxx.vmdk -d thin
```

Step 2 Execute `vim-cmd vmsvc/getallvms` to get the vmid of the VM where the disk needs to be added.

```
Vmid Name                      File                                Guest OS    Version Annotation
173  vminstaller-AIO [datastore5] vminstaller-AIO/vminstaller-AIO.vmx centos64Guest vmx-08
```

Step 3 Assign the disk to the VM.

The *xxxx* is the disk name, and 0 and 1 indicate the SCSI device number.

In this example, this is the second disk:

```
vim-cmd vmsvc/device.diskaddexisting vmid /vmfs/volumes/path to xxxx.vmdk 0 1
```

Target VM Configuration

Step 1 Log in as root user on your Linux virtual machine.

Step 2 Open a terminal session.

Step 3 Execute the `df` command to examine the current disks that are mounted and accessible.

Step 4 Create an ext4 file system on the new disk:

```
mkfs -t ext4 /dev/sdb
```

Note `b` in `/dev/sdb` is the second SCSI disk. It warns that you are performing this operation on an entire device, not a partition. That is correct, since you created a single virtual disk of the intended size. This is assuming you have specified the correct device. Make sure you have selected the right device; there is no undo.

Step 5 Execute the following command to verify the existence of the disk you created:

```
# fdisk -l
```

Step 6 Execute the following command to create a mount point for the new disk:

```
# mkdir /<NewDirectoryName>
```

Step 7 Execute the following command to display the current `/etc/fstab`:

```
# cat /etc/fstab
```

Step 8 Execute the following command to add the disk to `/etc/fstab` so that it is available across reboots:

```
/dev/sdb /<NewDirectoryName> ext4 defaults 1 3
```

Step 9 Reboot the VM.

```
shutdown -r now
```

Step 10 Execute the `df` command to check the file system is mounted and the new directory is available.

Update the collectd process to use the new file system to store KPIs

After the disk is added successfully, **collectd** can use the new disk to store the KPIs.

Step 1 SSH into pcrfclient01/pcrfclient02.

Step 2 Execute the following command to open the logback.xml file for editing:

```
vi /etc/collectd.d/logback.xml
```

Step 3 Update the file element <*file*> with the new directory that was added in the `/etc/fstab`.

Step 4 Execute the following command to restart `collectd`:

```
monit restart collectd
```

Note The content of logback.xml will be overwritten to the default path after a new upgrade. Make sure to update it after an upgrade.

Mounting the Replication Set from Disk to tmpfs After Deployment

You can mount all of the members of the Replication set to tmpfs, or you can mount specific members to tmpfs. These scenarios are described in the following sections.

Scenario 1 – Mounting All Members of the Replication Set to tmpsf

Step 1 Modify `mongoConfig.cfg` file using the vi editor on cluster manager. Change the DBPATH directory for the SPR Replication set that needs to be put on tmpfs.

Note Make sure you change the path to `/var/data/sessions.1`, which is the tmpfs filesystem. Also, make sure to run diagnostics.sh before and after the activity.

The following example shows the contents of `mongoConfig.cfg` file before modification:

```
[SPR-SET1]
SETNAME=set06
OPLOG_SIZE=5120
ARBITER1=pcrfclient01a:27720
ARBITER_DATA_PATH=/var/data/sessions.6
MEMBER1=sessionmgr04a:27720
MEMBER2=sessionmgr03a:27720
MEMBER3=sessionmgr04b:27720
MEMBER4=sessionmgr03b:27720
DATA_PATH=/var/data/sessions.4
[SPR-SET1-END]
```

The following example shows the contents of `mongoConfig.cfg` file after modification:

```
[SPR-SET1]
SETNAME=set06
OPLOG_SIZE=5120
ARBITER1=pcrfclient01a:27720
ARBITER_DATA_PATH=/var/data/sessions.6
MEMBER1=sessionmgr04a:27720
MEMBER2=sessionmgr03a:27720
MEMBER3=sessionmgr04b:27720
MEMBER4=sessionmgr03b:27720
DATA_PATH=/var/data/sessions.1/set06
[SPR-SET1-END]
```

Step 2 Run `build_etc.sh` to update the modified files.

Step 3 Verify that the sessionmgr-27720 files on sessionmgr VMs are updated with new DB_PATH by using `vi` or `cat` command.

Step 4 Stop and start the mongo databases one by one using the following commands:

```
systemctl stop sessionmgr-<port>
```

```
systemctl start sessionmgr-<port>
```

Step 5 Run `diagnostics.sh`.

Step 6 If this is an Active/Active GEOHA setup, scp the `mongoConfig.cfg` file to Site-B Cluster Manager, and run `build_etc.sh` to update puppet files.

Scenario 2 – Mounting Specific Members of the Replication Set to tmpfs

Step 1 Ssh to the respective session manager.

Step 2 Edit the mongoDB startup file using the vi editor. In this example we are modifying the SPR member.

```
[root@sessionmgr01 init.d]# vi /etc/init.d/sessionmgr-27720
```

Step 3 Change the DBPATH directory from `DBPATH=/var/data/sessions.4` to `DBPATH=/var/data/sessions.1/set06`.

Step 4 Save and exit the file (using !wq).

Step 5 Enter the following commands to stop and start the SPR DB member:

```
/usr/bin/systemctl stop sessionmgr-27720
/usr/bin/systemctl start sessionmgr-27720
```

Step 6 Wait for the recovery to finish.

Manage Disks to Accommodate Increased Subscriber Load

If you need to prepare CPS for an increased number of subscribers (> 10 million), you can clone and repartition the sessionmgr disks as per your requirement.

Clone Sessionmgr01 VM

Downtime: No downtime

Before you begin

- Before disk repartition, clone sessionmgr01. This step is optional but to reduce the risk of losing the data during disk repartitioning, the customer can take the backup of sessionmgr01 VM. If the customer does not have enough space to take the backup this step can be ignored.
- Blade with enough space to hold cloned image of sessionmgr01.

Step 1 Login to vSphere Client on sessionmgr01 blade with administrator credentials.

Step 2 Right-click sessionmgr01 and select **Clone** > Choose appropriate inventory in which blade resides > Choose the blade with enough space to hold sessionmgr01 image > **Next** > **Next** > **Finish**.

Step 3 Cloning starts. Wait for it to finish the process.

Disk Repartitioning of Sessionmgr01 VM

Downtime: During this procedure Sessionmgr01 is shut down 2 times. Estimate approximately 30 minutes of downtime for sessionmgr01.

CPS continues to operate using the other sessionmgr02 while sessionmgr01 is stopped as part of procedure.

Before you begin

None

Step 1 Login to sessionmgr01 as a root user.

Step 2 The following commands may be executed to help identify which partition requires additional space.

```
 synph# df -h/synph
synphFilesystem                       Size  Used Avail Use% Mounted on/synph
synph/dev/mapper/vg_shiprock-lv_root  7.9G  1.5G  6.0G  20% //synph
synphtmpfs                            1.9G     0  1.9G   0% /dev/shm/synph
synph/dev/sda1                        485M   32M  428M   7% /boot/synph
synph/dev/mapper/vg_shiprock-lv_home  2.0G   68M  1.9G   4% /home/synph
synph/dev/mapper/vg_shiprock-lv_var    85G   16G   65G  20% /var/synph
synphtmpfs                            2.3G  2.1G  172M  93% /var/data/sessions.1/synph
synph/synph
synph# pvdisplay/synph
synph  --- Physical volume ---/synph
synph  PV Name               /dev/sda2/synph
synph  VG Name               vg_shiprock/synph
synph  PV Size               99.51 GiB / not usable 3.00 MiB/synph
synph  Allocatable           yes (but full)/synph
synph  PE Size               4.00 MiB/synph
synph  Total PE              25474/synph
synph  Free PE               0/synph
synph  Allocated PE          25474/synph
synph  PV UUID               l3Mjox-tLfK-jj4X-98dJ-K3c1-EOel-SlOBq1/synph
synph/synph
synph# vgdisplay/synph
synph--- Volume group ---/synph
synph  VG Name               vg_shiprock/synph
synph  System ID             /synph
synph  Format                lvm2/synph
synph  Metadata Areas        1/synph
synph  Metadata Sequence No  5/synph
synph  VG Access             read/write/synph
synph  VG Status             resizable/synph
synph  MAX LV                0/synph
synph  Cur LV                4/synph
synph  Open LV               4/synph
synph  Max PV                0/synph
synph  Cur PV                1/synph
synph  Act PV                1/synph
synph  VG Size               99.51 GiB/synph
synph  PE Size               4.00 MiB/synph
synph  Total PE              25474/synph
synph  Alloc PE / Size       25474 / 99.51 GiB/synph
synph  Free  PE / Size       0 / 0   /synph
synph  VG UUID               P1ET44-jiEI-DIbd-baYt-fVom-bhUn-zgs5Fz/synph
```

- (df -h): /var is /dev/mapper/vg_shiprock-lv_var. This is equivalent to device /dev/vg_shiprock/lv_var.
- (pvdisplay): vg_shiprock (used by lv_var which is /var) is on /dev/sda2.

Step 3 Execute the fdisk command to check the disk size.

```
# fdisk -l /dev/sda

Disk /dev/sda: 107.4 GB, 107374182400 bytes
255 heads, 63 sectors/track, 13054 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0008dcae

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          64      512000   83  Linux
Partition 1 does not end on cylinder boundary.
/dev/sda2              64       13055   104344576   8e  Linux LVM
```

Step 4 Power down the Virtual Machine.

```
# shutdown -h now
```

Note If cloning is not possible because of space limitation on Blade, backup of sessionmgr01 VM can be taken by saving OVF of sessionmgr01 VM to local storage like Laptop, Desktop. (Both cloning and OVF backup are optional steps, but either one of them is highly recommended.)

Step 5 Log in using the VMware vSphere Client as an administrator (e.g. root) to the ESXi host which has your Linux Virtual Machine on it.

Step 6 Right-click on the Virtual Machine and select Edit Settings > Click Hard Disk 1 > Increase the Provisioned Size of the Hard Disk.

Step 7 Power ON the Virtual Machine.

Step 8 Login (ssh) to the Virtual Machine as root user.

Step 9 Confirm that disk space has been added to the /dev/sda partition.

```
# fdisk -l /dev/sda

Disk /dev/sda: 70.5 GB, 79529246720 bytes
255 heads, 63 sectors/track, 9668 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

Step 10 Execute the following commands (Bold Characters indicates actual inputs from user (all of them are in lower case).

```
# fdisk /dev/sda
The number of cylinders for this disk is set to 7832.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)
Command (m for help): p
Disk /dev/sda: 64.4 GB, 64424509440 bytes
255 heads, 63 sectors/track, 7832 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1          13      104391   83  Linux
/dev/sda2              14        7179    57560895   8e  Linux LVM
Command (m for help): d
Partition number (1-4): 2
```

```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (14-7832, default 14):  [press enter]
Using default value 14
Last cylinder +sizeM/+sizeK (14-7832,default 7832): [press enter]
Using default value 7832
Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 8e
Changed system type of partition 2 to 8e (Linux LVM)
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

Step 11 Reboot the sessionmgr01 VM by executing the following command:

```
# reboot
```

This ensures that the new setting match up with the kernel.

Step 12 After reboot, execute following command:

```
# pvresize /dev/sda2
Physical volume "/dev/sda2" changed
1 physical volume(s) resized / 0 physical volume(s) not resized
```

Step 13 Confirm that the additional free space is added in sessionmgr VM.

```
# vgdisplay
  --- Volume group ---
  VG Name               vg_shiprock
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  5
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                4
  Open LV               4
  Max PV                0
  Cur PV                1
  Act PV                1
VG Size             129.51 GiB
PE Size              4.00 MiB
Total PE             32974
Alloc PE / Size     25474 / 99.51 GiB
Free PE / Size     7500 / 30.00 GB
VG UUID              pPSNBU-FRWO-z3aC-iAxS-ewaw-jOFT-dTcBKd
```

Step 14 Verify that the /var partition is mounted on /dev/mapper/vg_shiprock-lv_var.

```
#df -h
Filesystem           Size Used Avail Use% Mounted on
/dev/mapper/vg_shiprock-lv_root
      18G 2.5G    15G 15% /
/dev/mapper/vg_shiprock-lv_home
```

```
      5.7G 140M 5.3G   3% /home
/dev/mapper/vg_shiprock-lv_var
      85G   16G   65G   20% /var
/dev/sda1              99M   40M   55M  43% /boot
tmpfs                  16G     0   16G   0% /dev/shm
tmpfs                 8.0G  1.1G  7.0G  14% /data/sessions.1
```

Step 15 Extend /var partition to take up additional free space.

```
#lvextend -l +100%FREE /dev/mapper/vg_shiprock-lv_var
 Extending logical volume lv_var to 120.00 GB
 Logical volume lv_var successfully resized
```

Step 16 Check the newly added space in /dev/mapper/vg_shiprock-lv_var.

```
# lvdisplay
```

Step 17 Add space to VM file system.

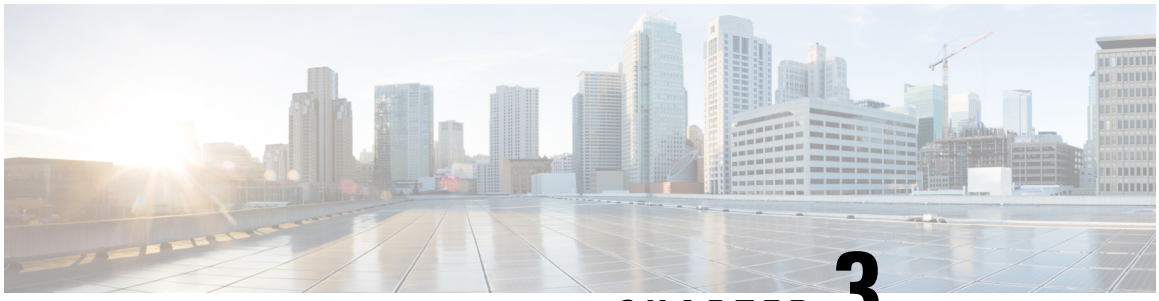
```
# resize2fs /dev/mapper/vg_shiprock-lv_var
resize2fs 1.39 (29-May-2006)
Filesystem at /dev/mapper/vg_shiprock-lv_var is mounted on /var; on-line resizing required
Performing an on-line resize of /dev/mapper/vg_shiprock-lv_var to 6553600 (4k) blocks.
The filesystem on /dev/mapper/vg_shiprock-lv_var is now 6553600 blocks long.
```

Step 18 Check the increased size of /var partition.

```
# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/vg_shiprock-lv_root
                       23G  2.1G   20G  10% /
/dev/mapper/vg_shiprock-lv_home
                      5.7G  140M  5.3G   3% /home
/dev/mapper/vg_shiprock-lv_var
                      130G   16G   95G  12% /var
/dev/sda1              99M   40M   55M  43% /boot
tmpfs                 2.0G     0  2.0G   0% /dev/shm
```

Cloning and Disk Repartitioning of Sessionmgr02 VM

Repeat Clone Sessionmgr01 VM, on page 16 and Disk Repartitioning of Sessionmgr01 VM, on page 17 on sessionmgr02 for cloning and disk repartitioning of sessionmgr02 VM.



CHAPTER 3

Managing CPS Licenses

- Smart Software Licensing, on page 21
- Classic Licensing, on page 21
- Comparison between Licensing Models, on page 22
- Smart Accounts/Virtual Accounts, on page 23
- License Conversion, on page 24
- Enable Smart Licensing for CPS, on page 25
- Product ID Tags, on page 27
- Smart Licensing CLI Commands, on page 27
- License Usage Threshold, on page 29

Smart Software Licensing

CPS 10.0.0 and its later releases support Smart Licensing. It is a cloud-based approach to licensing that simplifies the purchase, deployment, and management of Cisco software assets. Entitlements are purchased through your Cisco account via Cisco Commerce Workspace (CCW) and immediately deposited into your Virtual Account for usage. This eliminates the need to install license files on every device. Products that are smart enabled communicate directly to Cisco to report consumption. A single location is available to customers to manage Cisco software licenses—the Cisco Smart Software Manager (CSSM). License ownership and consumption are readily available to help make better purchase decision based on consumption or business need.

Classic Licensing

Classic Licensing is Cisco's legacy licensing model based on Product Activation Keys (PAK) and Unique Device Identifiers (UDI). On most IOS devices, a determination of bandwidth needs is assessed prior to obtaining and installing a tar file on the platform to retrieve the UDI. A PAK is ordered and typically emailed to the customer. The combination of a UDI and PAK are used to receive a license file, which is installed in the boot directory to complete the installation of IOS on the platform. The License Registration Portal (LRP) is available to help migrate Classic Licenses to Smart Licenses. To access the LRP, and to obtain training and manage licenses, visit http://tools.cisco.com/SWIFT/LicensingUI/Home.

Comparison between Licensing Models

The following sections provide a comparison of the existing CPS SWIFT-based licensing model, the Cisco Smart Software Licensing model, and Cisco Smart Software Licensing as it is implemented in CPS 10.0.0 and later releases.

CPS SWIFT-Based Licensing

For CPS versions prior to 10.0.0, CPS licensing is SWIFT "lmgrd" based, and the license is tied to the MAC address of the device on which CPS is installed. The following list summarizes the CPS SWIFT-based licensing model:

- The License count that is purchased by the customer is defined in the `license.lic` file and is read into the CPS application using the lmgrd/cisco processes.
- License compliance is determined and tracked by CPS. CPS periodically compares the current session count with the licensed count at a predefined interval.
- CPS creates and logs license statuses: adhere, "RATE_LIMITED" and "VALID" statuses are logged with proper messages, and traps are generated accordingly.

Cisco Smart Software Licensing

The following list summarizes the Cisco Smart Software License model:

- Smart Licensing maintains and tracks license information including license quantity, license surplus, and shortage usages.
- There is no API for returning the number of licenses (entitlements) purchased by the customer.
- License compliance is determined and tracked by Cisco Smart Software License. Entitlement enforcement mode notifications will send out when it is changed upon the request.
- License (entitlement) expiration is tracked by Cisco Smart Software License. There is no API for returning the license expiration date.
- Smart Licensing does not support license version.
- Utility/Metering is not supported.
- An entitlement consumption request is allowed once every 24 hours maximum.
- Smart Licensing supports high availability. For Smart Agent clusters, one Smart Agent is active and the rest are standbys. This means that for a given cluster, only one Smart Agent is active, and it will register to the Smart Licensing portal at any time. (Smart License is a combination of Smart Agent and Smart Call Home, which is responsible for communicating to Cisco Smart Software Licensing.)

CPS Cisco Smart Software License Based Model

The following list summarizes the Cisco Smart Software License model for CPS 10.0.0 and greater:

- For a CPS high availability installation, only the active client (either pcrfclient01 or pcrfclient02) is registered to the Smart Licensing Portal at any given time, and it uses the same identify for the registration.

- CPS uses the Smart Licensing API to request the entitlement (license) consumption amount based on the pre-defined maximum licensed concurrent session amount.
- The predefined maximum licensed concurrent session amount is defined in the `features.properties` file for each CPS feature.
- One licensed entitlement count is equivalent to one CPS Policy concurrent session count.
- Smart Licensing Entitlement notifies CPS about the requested entitlement conformance (enforce mode) if the requested entitlement consumption is InCompliance or OutCompliance or Eval, meaning that the product instance is not registered to the Smart Licensing Portal and is running in evaluation mode. CPS populates license data into mongoDB: `shardding/licensedfeats <SITEID>` collection based on the received entitlement compliance status.
- The Smart Agent (SA) is embedded in CPS+SL (SA+SCH) integration. A CLI is supported.
- CPS Orchestration API-based installation is not supported.
- Dynamically switching the license manager from lmgrd to Smart Licensing or vise versa is supported. Switching the licensing manager requires a restart of CPS OAM (pcrfclient).
- CPS Smart Licensing integration follows the CPS In-Service Software Upgrade process.

In summary, CPS 10.0.0 and later releases support the same functionality as CPS SWIFT imgrd-based licensing with the following exceptions:

- There is no API to return the license amount available for the virtual account. A new “complianceMode” attribute has been added to indicate the requested feature entitlement compliance status with the following value options:
  - InCompliance – The requested feature entitlement maximum licensed amount is in surplus status.
  - OutOfCompliance – The requested feature entitlement maximum licensed amount is in shortage status.
  - Eval – The product is not yet registered to Cisco Smart License Cloud.
- There is no API to return the license expiration date. The license expiration date value will set to “current date + 10 years future date” in CPS 10.0.0 and later releases.
- Smart Licensing does not support license version. Currently, the license version is set to “V1.0” in CPS.

Smart Accounts/Virtual Accounts

A Smart Account provides a single location for all Smart-enabled products and entitlements. It helps speed procurement, deployment, and maintenance of Cisco Software. When creating a Smart Account, you must have the authority to represent the requesting organization. After submitting, the request goes through a brief approval process.

A Virtual Account exists as a sub-account withing the Smart Account. Virtual Accounts are a customer-defined structure based on organizational layout, business function, geography or any defined hierarchy. They are created and maintained by the Smart Account administrator.

See http://software.cisco.com to learn about, set up, or manage Smart Accounts.

Request a Cisco Smart Account

A Cisco Smart Account is an account where all products enabled for Smart Licensing are deposited. A Cisco Smart Account allows you to manage and activate your licenses to devices, monitor license use, and track Cisco license purchases. Through transparent access, you have a real-time view into your Smart Licensing products. IT administrators can manage licenses and account users within your organization's Smart Account through the Smart Software Manager.

Step 1 In a browser window, enter the following URL:

```
http://software.cisco.com
```

Step 2 Log in using your credentials, and then click **Request Smart Account** in the **Administration** area under **Smart Account Management**.

The **Smart Account Request** window is displayed.

Step 3 Under **Create Account**, select one of the following options:

- **Yes, I have authority to represent my company and want to create the Smart Account** – If you select this option, you agree to authorization to create and manage product and service entitlements, users, and roles on behalf of your organization.
- **No, the person specified below will create the account** – If you select this option, you must enter the email address of the person who will create the Smart Account.

Step 4 Under **Account Information**:

a) Click **Edit** beside **Account Domain Identifier**.

b) In the **Edit Account Identifier** dialog box, enter the domain, and click **OK**. By default, the domain is based on the email address of the person creating the account and must belong to the company that will own this account.

c) Enter the **Account Name** (typically, the company name).

Step 5 Click **Continue**.

The Smart Account request will be in pending status until it has been approved by the Account Domain Identifier. After approval, you will receive an email confirmation with instructions for completing the setup process.

Cisco Smart Software Manager

Cisco Smart Software Manager (CSSM) enables the management of software licenses and Smart Account from a single portal. The interface allows you to activate your product, manage entitlements, and renew and upgrade software. A functioning Smart Account is required to complete the registration process. To access the Cisco Smart Software Manager, see https://software.cisco.com/.

License Conversion

Using the License Registration Portal, you can convert classic licenses that are associated with Product Activation Keys (PAKs) to smart entitlements.

Step 1 To access the License Registration Portal:

a) Login to the **Cisco Software Central** page at software.cisco.com.

b) Under **License**, click **Traditional Licensing**.

On the **Welcome to the Product License Registration Portal** window, you can choose to watch training videos, or you can go directly to the Product License Registration Portal.

c) Select the **Product License Registration Portal** option.

The **Product License Registration** page opens.

Step 2 Select the **PAKs/Tokens** tab to access your classic licenses.

Step 3 On the **PAKs/Tokens** tab, check the box next to the PAK/Token ID for which you want to convert licenses.

Step 4 From the **Actions** drop-down list, select **Convert to Smart Entitlements**.

In the **Convert to Smart Entitlements** dialog box, you can change to a different Virtual Account if needed.

Step 5 Check the box next to the PAK.

Step 6 Enter the **Quantity to Convert**, and click **Submit**.
You will receive a message when the conversion has completed successfully.

Step 7 Login to the Cisco Smart Software Manager (CCSM), and view the converted Smart Entitlements as follows:

a) Select the Virtual Account, and click the **License Conversion** tab.

b) Click the **Event Log** tab to see the confirmation message that the licenses were converted.

Enable Smart Licensing for CPS

You can enable smart licensing after upgrading CPS, or after a new CPS deployment.

Note These steps must be performed on the Cluster Manager VM.

Step 1 Log in to the Cluster Manager VM.

Step 2 Enter the following commands to create `license_sl_data` and `license_sl_conf` directories:

```
mkdir -p /etc/broadhop/license_sl_data
mkdir -p /etc/broadhop/license_sl_conf
```

Step 3 Create the following license configuration files in the `/etc/broadhop/license_sl_conf` directory on the Cluster Manager:

a) Create a file named `features.properties`, and add the required PID and count. For example:

```
LicenseFeature=<PID>:<COUNT>
```

b) Create a file named `sl.properties` with the following content from the CSSM account:

```
TRANSPORT_URL=https://tools.cisco.com/its/service/oddce/services/DDCEService
```

c) Create a file named `conf.properties` with the following content from the CSSM account. For example:

```
PRODUCT_SN=10999
PRODUCT_ID_TAG=CPS
SOFTWARE_ID_TAG=regid.2016-06.com.cisco.CPS10,1.0_e454cefa-5e10-4af4-81d8-3f76260485fb
USE_PROD_ROOT_CERT=true
RENEW_AUTH=false
TAC_PROFILE_NAME=CiscoTAC-1
HTTP_TRANSPORT_FLAG=true
HTTP_URL=https://tools.cisco.com/its/service/oddce/services/DDCEService
PRODUCT_NAME=Cisco Policy Suite
SOFTWARE_VERSION=10.0
SYSTEM_DESCRIPTION=Cisco Policy Suite for Mobile is a carrier-grade policy, charging, and
subscriber data management solution.
PRODUCT_SERIES=Cisco Policy Suite Series
```

Step 4 Enter the following command to rebuild the `/etc/broadhop/license_sl_data` and `license_sl_conf` directory in the Cluster Manager VM:

```
/var/qps/install/current/scripts/build/build_etc.sh
```

Step 5 Enter the following commands to push the license to pcrfclient01 and pcrfclient02:

```
ssh pcrfclient01
   /etc/init.d/vm-init

ssh pcrfclient02
   /etc/init.d/vm-init
```

Step 6 Enter the following commands to map the Smart License server hostname to the IP address and to synchronize the `/etc/hosts` files across the VMs:

```
echo "173.37.145.8 tools.cisco.com" >> /etc/hosts
/var/qps/bin/update/synchosts.sh
```

Step 7 Configure CPS to use Smart Licensing as follows:

a) Open the `qns.conf` file by entering the following command:

```
vi /etc/broadhop/qns.conf
```

b) Edit the `qns.conf` file, and add the following argument:

```
-Dcom.broadhop.license.approach=sl
```

c) Save and close the `qns.conf` file.

d) Enter the following commands to copy the modified `qns.conf` file from Cluster Manager to all of the VMs:

```
copytoall.sh /etc/broadhop/qns.conf /etc/broadhop/qns.conf
restartall.sh
```

Step 8 To view license related logs, see the following log file:

```
/var/log/broadhop/license.log
```

Step 9 Access the Cisco Smart Software Manager (CSSM) at the following location:

https://software.cisco.com/

Step 10 Select the appropriate virtual account, and then click **New Token** in the **General** tab.

Step 11 In the **Create Token** dialog box, enter the required information, accept the terms and responsibilities, and then click **Create Token**.

Step 12 Select the token text, and copy it to your clipboard.

Step 13 Enter the following command, pasting the token that you copied in place of <*token*>:

```
license smart register idtoken <token> [force]
```

Product ID Tags

Tags for the following PIDs have been created to enable the proper product IDs to be identified, reported, and enforced.

Table 3: PID Tags

PID	Entitlement Tag	Entitlement name in CSSM
POLICY-VALUE	regid.2016-06.com.cisco.POLICY-VALUE, 1.0_7f667e53-11e1-40e2-9480-ff7eb064561c	CPS Value Plus Feature Pack
POLICY-ALL	regid.2016-06.com.cisco.POLICY-ALL, 1.0_65566461-0788-4c92-8ffa-f9a02e9843e8	CPS All Inclusive Feature Pack
POLICY-UPGRADE	regid.2016-06.com.cisco.POLICY-UPGRADE, 1.0_8fa236bc-e481-4673-aa4d-7da8f707647c	CPS Upgrade from Value Plus to All Inclusive Feature Pack
POLICY-ADD	regid.2016-06.com.cisco.PCRF-ADD, 1.0_676d51ca-4e14-40b3-81e7-1a600d726ce7	CPS PCRF Application License - Additional Applications

Smart Licensing CLI Commands

The following sections describe the commands that you can use to register, view information for, and manage Smart Licenses on your CPS systems.

Note These commands must be run on the active pcrfclient.

Register your Smart License

You must issue the following command to register your Smart License:

```
license smart register idtoken <token> [force]
```

This command registers the device with Cisco using an ID token that you obtain from the CSSM. The agent will register this product with Cisco and receive back an identity certificate. This certificate is saved and automatically used for all future communications with Cisco. After registration it will send the current license usage information to Cisco. Every 180 days the agent will automatically renew the registration information with Cisco. The ID token is not saved on the device.

This only needs to be done once per device.

The force option will cause the device to attempt registration even if it thinks it is already registered.

Show Smart License Information

You can use the following commands to view information related to your Smart License:

- `show license status`
- `show license summary`
- `show license UDI`
- `show license usage`
- `show license all`
- `show license tech support`

Manage your Smart License

You can use the following commands to manage your Smart License:

- `license smart renew ID`

  Dependency – Before using this command, Smart Licensing must be registered using the **license smart register idtoken** command.

  This command initiates a manual update of the license registration information with Cisco. Since the registration renewal is automatically done by the agent every 6 months, the customer will probably never need to use this command. It is available if for some reason the user needs to renew the registration information manually.

- `license smart renew auth`

  Dependency – Before using this command, Smart Licensing must have been registered using the **license smart register idtoken** command.

  This command manually refreshes license authorization information with Cisco. Since the license authorization is renewed automatically by the agent every 30 days, the customer will probably never need to use this command. It is available if for some reason the user needs to renew the license authorization information manually.

- `license smart deregister`

  Dependency – Before using this command, Smart Licensing must have been registered using the **license smart register idtoken** command.

  This command unregisters the device. The agent will try to contact the Cisco licensing cloud and unregister itself. All Smart Licensing entitlements and certificates on the platform will be removed. All certificates and registration information will be removed from the trusted store. This is true even if the agent is unable to communicate with Cisco to unregister. If the customer wants to use Smart Licensing again, they must run the **license smart register idtoken** command again.

License Usage Threshold

The Fault list configuration in Policy Builder allows configuring the thresholds at which License Usage Threshold Exceeded traps are sent out. The default recommended values are: Critical 95, Major 90, Minor 85 and Warning 80 which would result in traps being sent at 80, 85, 90 and 95 percent for License Usage Threshold Limits.

For example, if the license limit is 10000 sessions and there are 9600 active sessions, configuring the threshold at 95 and type as Critical would generate a Critical trap whose message is Session Count License Usage at 96%, exceeding threshold: 95%.

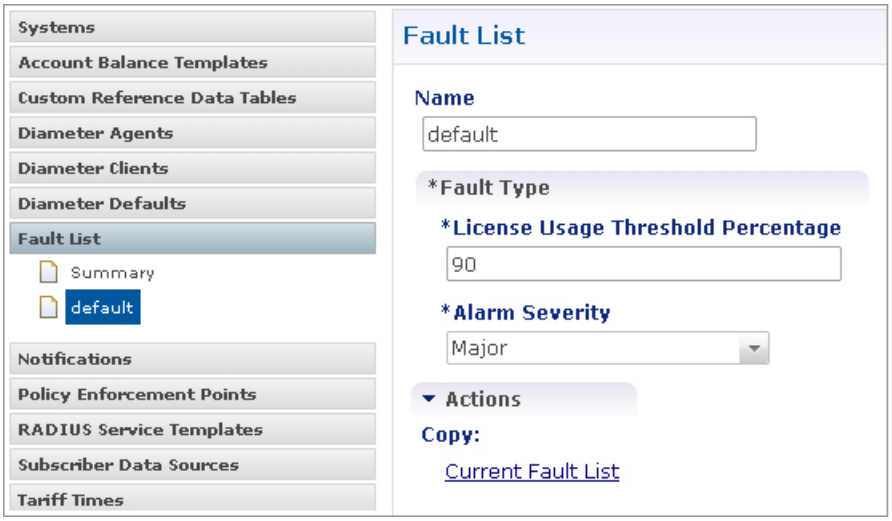
Configuration

Step 1 Open the Policy Builder GUI.

Step 2 Go to **Reference Data** tab and select **Fault List** from the left pane.

Step 3 Under **Create Child**, click **Fault List** to create a **License Usage Threshold Fault** as below.

Figure 1: License Usage Threshold Fault



Step 4 Choose a **Name** for the Fault List. Currently, only License Usage Threshold Percentage fault type is supported. The **Alarm Severity** can be configured to be one of **Critical**, **Major**, **Minor** or **Warning**.

The recommended values for License Usage Threshold Percentage are:

- Critical 95
- Major 90
- Minor 85
- Warning 80

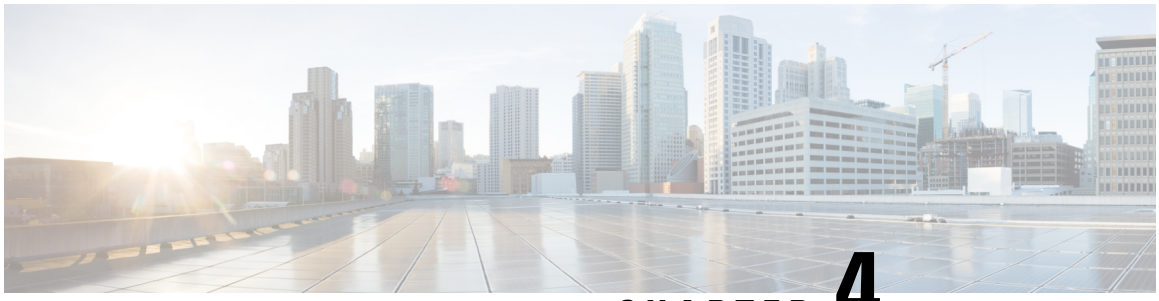
The above PB configuration when saved and published would trigger an application trap of type MAJOR when the 90% threshold configuration is crossed. One example of a trap sent would be number of licenses exceeded.

Validation Steps

Step 1 Configure a Threshold Limit as explained above in PB.

Step 2 Generate active sessions exceeding the configured threshold limit.

Step 3 Validate the Traps are received on the configured trap receiver for the defined limit and Severity.



CHAPTER 4

Managing CPS Interfaces and APIs

- CPS Interfaces and APIs, on page 31
- Multi-user Policy Builder, on page 53
- Control Center Access, on page 55
- Enabling Authentication and Authorization for CRD API, on page 59
- Unified API Security: Access Privileges, on page 61
- Enabling Unified API Access on HTTP Port 8080, on page 63
- TACACS+, on page 66
- CRD APIs, on page 69

CPS Interfaces and APIs

CPS includes southbound interfaces to various policy control enforcement functions (PCEFs) in the network, and northbound interfaces to OSS/BSS and subscriber applications, IMSs, and web applications.

Control Center GUI Interface

Purpose

Cisco Control Center enables you to do these tasks:

- Manage subscriber data, that is, find or create and edit information about your subscribers.
- View subscriber sessions.
- View system sessions.
- Populate custom reference data (CRD) tables.

URL and Port

HA: https://<*lbvip01*>:443

AIO: http://<*ip*>:8090

Protocol

HTTPS/HTTP

Accounts and Roles

There are two levels of administrative roles supported for Control Center: Full Privilege and View Only. The logins and passwords for these two roles are configurable in LDAP or in /etc/broadhop/authentication-password.xml.

- Full Privilege Admin Users: These users can view, edit, and delete information and can perform all tasks. Admin users have access to all screens in Control Center.
- View Only Admin Users: These users can view information in Control Center, but cannot edit or change information. View only administrators have access to a subset of screens in the interface.

CRD REST API

Purpose

The Custom Reference Data (CRD) REST API enables the query of, creation, deletion, and update of CRD table data without the need to access the Control Center GUI. The CRD APIs are available using an HTTP REST interface. The specific APIs are outlined in a later section in this guide.

URL and Port

HA: https:// <*lbvip01*>:443/custrefdata

AIO: http://<*ip*>:8080/custrefdata

A validation URL is:

HA: https:// <*lbvip01*>:8443/custrefdata

AIO: http://<*ip*>:8080/custrefdata

Protocol

HTTPS/HTTP

Accounts and Roles

Security and account management is accomplished by using the haproxy mechanism on the platform Policy Director (LB) by defining user lists, user groups, and specific users.

On Cluster Manager: `/etc/puppet/modules/qps/templates/etc/haproxy/haproxy.cfg`

Configure HAProxy

Update the HAProxy configuration to add authentication and authorization mechanism in the CRD API module.

1. Back up the `/etc/haproxy/haproxy.cfg` file.
2. Edit /etc/haproxy/haproxy.cfg on lb01/lb02 and add a userlist with at least one username and password as shown:

   ```
   userlist <userlist name>
   user <username1> password <encrypted password>
   ```

   For example:

```
userlist cps_user_list
user readonly password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
user apiuser password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
```

Run the following command to generate an encrypted password:

```
/sbin/grub-crypt --sha-512
```

For example:

```
[root@host ~]# /sbin/grub-crypt --sha-512
Password:
Retype password:
<encrypted password output>
```

3. Add the following line in frontend `https-api` to enable Authentication and Authorization for CRD REST API and create a new backend server as `crd_api_servers` to intercept CRD REST API requests:

```
mode http
acl crd_api path_beg -i /custrefdata/
use_backend crd_api_servers if crd_api
backend crd_api_servers
       mode http
       balance roundrobin
       option httpclose
       option abortonclose
       server qns01_A qns01:8080 check inter 30s
       server qns02_A qns02:8080 check inter 30s
```

4. Update frontend `https_all_servers` by replacing `api_servers` with `crd_api_servers` for CRD API as follows:

```
acl crd_api path_beg -i /custrefdata/

use_backend crd_api_servers if crd_api
```

5. Edit `/etc/haproxy/haproxy.cfg` on lb01/lb02 as follows:
   1. Add at least one group with user in userlist created in *Step 2* as follows:

      ```
      group qns-ro users readonly

      group qns users apiuser
      ```

   2. Add the following lines to the `backend crd_api_servers`:

      ```
      acl authoriseUsers http_auth_group(<cps-user-list>) <user-group>

      http-request auth realm CiscoApiAuth if !authoriseUsers
      ```

   Map the group created in *Step 5* with the acl as follows:

   ```
   acl authoriseUsers http_auth_group(<cps-user-list>) <user-group>
   ```

6. Add the following in the `backend crd_api_servers` to set read-only permission (GET HTTP operation) for group of users:

```
http-request deny if !METH_GET authoriseUsers
```

HAProxy Configuration Example

```
userlist cps_user_list
       group qns-ro users readonly
       group qns users apiuser
     user readonly password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75B

XKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
     user apiuser password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75B

XKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
frontend https-api
   description API
   bind lbvip01:8443 ssl crt /etc/ssl/certs/quantum.pem

   mode http
   acl crd_api path_beg -i /custrefdata/
   use_backend crd_api_servers if crd_api

   default_backend api_servers
   reqadd X-Forwarded-Proto:\ https if { ssl_fc }

frontend https_all_servers
description Unified API,CC,PB,Grafana,CRD-API,PB-AP
bind lbvip01:443 ssl crt /etc/ssl/certs/quantum.pem no-sslv3 no-tlsv10
ciphers ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES:RSA+AESGCM:RSA+AES:!
aNULL:!eNULL:!LOW:!     3DES:!MD5:!EXP:!PSK:!SRP:!DSS
mode http
acl crd_api path_beg -i  /custrefdata/
use_backend crd_api_servers if crd_api
backend crd_api_servers
       mode http
       balance roundrobin
       option httpclose
       option abortonclose
       server qns01_A qns01:8080 check inter 30s
       server qns02_A qns02:8080 check inter 30s
       acl authoriseReadonlyUsers http_auth_group(cps_user_list) qns-ro
       acl authoriseAdminUsers http_auth_group(cps_user_list) qns
       http-request auth realm CiscoApiAuth if !authoriseReadonlyUsers
!authoriseAdminUsers
       http-request deny if !METH_GET authoriseReadonlyUsers
```

Note The haproxy.cfg file is generated by the Puppet tool. Any manual changes to the file in lb01/lb02 would be reverted if the pupdate or vm-init scripts are run.

Grafana

Purpose

Grafana is a metrics dashboard and graph editor used to display graphical representations of system, application KPIs, bulkstats of various CPS components.

URL and Port

HA: https://<*lbvip01*>:9443/grafana

AIO: http://<*ip*>:443/grafana

Protocol

HTTPS/HTTP

Accounts and Roles

In CPS 7.5 and higher, at least one Grafana user account must be created to access the Grafana web interface.

In CPS 8.1 and higher, an administrative user account must be used to add, modify, or delete Grafana dashboards or perform other administrative actions.

Refer to the *Graphite and Grafana* chapter in this guide for details on adding or deleting these user accounts.

HAProxy

Purpose

Haproxy is a frontend IP traffic proxy process in lb01/lb02 that routes the IP traffic for other applications in CPS. The details of individual port that haproxy forwards is already described in other individual sections.

As per the Diameter configuration done, haproxy-diameter statistics will bind to one of the configurations and that URL will be displayed in `about.sh` output. For various options for Diameter configuration, refer to *Diameter Related Configuration* section in *CPS Installation Guide for VMware*.

More information about HAProxy is provided in the HAProxy, on page 134.

Documentation for HAProxy is available at: http://www.haproxy.org/#docs

URL and Port

To view statistics, open a browser and navigate to the following URL:

- **For HAProxy Statistics**: http://<diameterconfig>:5540/haproxy?stats
- **For HAProxy Diameter Statistics**: http://<diameterconfig>:5540/haproxy-diam?stats

Accounts and Roles

Not applicable.

JMX Interface

Purpose

Java Management Extension (JMX) interface can be used for managing and monitoring applications and system objects.

Resources to be managed / monitored are represented by objects called managed beans (mbeans). MBean represents a resource running in JVM and external applications can interact with mbeans through the use of

JMX connectors and protocol adapters for collecting statistics (pull); for getting/setting application configurations (push/pull); and notifying events like faults or state changes (push).

CLI Access

External applications can be configured to monitor application over JMX. In addition to this, there are scripts provided by application that connects to application over JMX and provide required statistics/information.

Port

pcrfclient01/pcrfclient02:

- Control Center: 9045
- Policy Builder: 9046

lb01/lb02:

- iomanager: 9045
- Diameter Endpoints: 9046, 9047, 9048...

qns01/qns02/qns... : 9045

Ports should be blocked using firewall to prevent access from outside the CPS system.

Accounts and Roles

Not applicable.

Logstash

Purpose

Logstash is a process that consolidates the log events from CPS nodes into pcrfclient01/pcrfclient02 for logging and alarms. The logs are forwarded to CPS application to raise necessary alarms and the logs are stored at `/var/log/logstash/logstash.log`.

Note If logstash in not monitoring, then check the Policy Server (qns) process using `monit summary`.

```
[root@dc1-pcrfclient01 ~]# monit summary
The Monit daemon 5.17.1 uptime: 18h 41m

Process 'whisper'                   Running
Process 'sssd'                      Running
Process 'snmpd'                     Running
Program 'kpi_trap'                  Status ok
Program 'db_trap'                   Status failed
Program 'failover_trap'             Status ok
Program 'qps_process_trap'          Status ok
Program 'admin_login_trap'          Status ok
Program 'vm_trap'                   Status ok
Program 'qps_message_trap'          Status ok
Program 'ldap_message_trap'         Status ok
Program 'logstash_process_status'   Status ok
Process 'qns-2'                     Running
Process 'qns-1'                     Does not exist
Process 'corosync'                  Running
Program 'monitor_replica'           Status ok
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'logstash'                  Not monitored
Program 'mon_db_for_lb_failover'    Status ok
Program 'mon_db_for_callmodel'      Status ok
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'carbon-cache'              Running
Process 'carbon-aggregator'         Running
Process 'auditrpms.sh'              Running
System 'dc1-pcrfclient01'           Running
```

On pcrfclient node, if Policy Server (qns) process is not running, 'logstash_process_status' program stops the logstash process so that the alarm is raised from another pcrfclient node.

CLI Access

There is no specific CLI interface for logstash.

Protocol

TCP and UDP

Ports

TCP: 5544, 5545, 7546, 6514

UDP: 6514

Accounts and Roles

Not applicable.

LDAP SSSD

Purpose

In CPS 14.0.0 and higher releases, SSSD based authentication is supported, allowing users to authenticate against an external LDAP server and gain access to the CPS CLI. SSSD RPMs and default `sssd.conf` file is installed on each CPS VM when you perform a new installation or upgrade CPS.

For more information, refer to the *CPS Installation Guide for VMware*.

`/etc/monit.d/sssd` file has been added with the following content so that SSSD is monitored by monit:

```
check process sssd with pidfile /var/run/sssd.pid
start program = "/etc/init.d/sssd start" with timeout 30 seconds
stop program = "/etc/init.d/sssd stop" with timeout 30 seconds
```

Also `/etc/logrotate.d/sssd` file has been added to rotate the SSSD log files. Here is the default configuration:

```
"
/var/log/sssd/*.log {
    daily
    missingok
    notifempty
    sharedscripts
    nodateext
    rotate 5
    size 100M
    compress
    delaycompress
    postrotate
        /bin/kill -HUP `cat /var/run/sssd.pid 2>/dev/null` 2> /dev/null || true
    endscript
}
"
```

Use the `monit summary` command to view the list of services managed by monit. Here is an example:

```
monit summary
The Monit daemon 5.17.1 uptime: 4d 2h 22m

Process 'whisper'                   Running
Process 'sssd'                      Running
Process 'snmptrapd'                 Running
Process 'snmpd'                     Running
Program 'vip_trap'                  Status ok
Program 'gr_site_status_trap'       Status ok
Process 'redis'                     Running
Process 'qns-4'                     Running
Process 'qns-3'                     Running
Process 'qns-2'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-4'                Accessible
File 'monitor-qns-3'                Accessible
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'memcached'                 Running
Process 'irqbalance'                Running
Process 'haproxy-diameter'          Running
Process 'haproxy'                   Running
Process 'cutter'                    Running
Process 'corosync'                  Running
```

```
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'lb01'                       Running
```

Important Setting of other configuration files to support LDAP based authentication and the changes required in `sssd.conf` file as per the customer deployment is out of scope of this document. For more information, consult your Cisco Technical Representative.

Restriction Grafana support LDAP authentication over httpd and does not use SSSD feature. Due to this, if LDAP server is down then grafana is not accessible for LDAP users.

CLI Access

No CLI is provided.

Port

Port number is not required.

Configure Policy Builder

Step 1 To provide admin access, enter username in the following file:

`/var/www/svn/users-access-file`

Note This action should be performed on pcrfclient and not on policy server (qns).

```
[groups]
admins = qns,qns-svn,sssd_pb_2
nonadmins = qns-ro
[/]
@admins = rw
@nonadmins = r
* = r
```

Step 2 Verify if you can export CRD data from the following link:

http://<aio_server>:7070/central/

Configure Grafana

Step 1 Bypass the first level authentication by updating the `/etc/httpd/conf.d/grafana-proxy.conf` file as follows:

```
LoadModule proxy_module modules/mod_proxy.so
LoadModule proxy_balancer_module modules/mod_proxy_balancer.so
LoadModule proxy_ftp_module modules/mod_proxy_ftp.so
LoadModule proxy_http_module modules/mod_proxy_http.so
LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule proxy_connect_module modules/mod_proxy_connect.so
# Set root to <ip address>/grafana
ProxyPass /grafana http://127.0.0.1:3000
ProxyPassReverse /grafana http://127.0.0.1:3000
# Set authentication for Grafana
# 1) Use httpd authentication as a front-end to Grafana
# 2) Remove header since Grafana is configured for anonymous
# authentication and will fail with a pass-thru header
#
# Notice: scope of authentication and header is limited to Grafana
# to avoid conflicts with other applications. Apache configuration
# in this file is global unless contained in the directive below.
<Location "/grafana">
    LoadModule headers_module modules/mod_headers.so
    Header set Access-Control-Allow-Origin "*"
    Header set Access-Control-Allow-Methods "GET, OPTIONS"
    Header set Access-Control-Allow-Headers "origin, authorization, accept"
    Header set Access-Control-Allow-Credentials true
    # Do not pass credentials to Grafana's anonymous authorization
    RequestHeader unset Authorization
    Satisfy Any
    #AuthName "Authentication Required"
    #AuthUserFile "/var/broadhop/.htpasswd"
    #Require valid-user
    #Order allow,deny
    # This is used for local calls to the API during puppet bring up
    Allow from 127.0.0.1
    #Satisfy Any
</Location>
```

Step 2 Restart httpd by running the following command:

```
/usr/bin/systemctl restart httpd
```

If `port already in use error` is displayed, execute the following steps:

a) Run the following command to get process ID:

```
ps -eaf | grep httpd
```

b) Run the following command to kill the pid:

```
kill -9 <pid>
```

Step 3 Update `/etc/grafana/grafana.ini` file to point to LDAP authentication instead of Basic Auth as follows:

```
################################### Basic Auth ##########################
[auth.basic]
# For CPS, trusted API requests come here and need local authentication
;enabled = true
################################### Auth LDAP ##########################
[auth.ldap]
enabled = true
config_file = /etc/grafana/ldap.toml
```

Step 4 Modify `/etc/grafana/ldap.toml` file to provide LDAP details (for example, search base dn, bind dn, group search base dn, member_of attribute) as follows:

```
# Set to true to log user information returned from LDAP
verbose_logging = true
```

```
[[servers]]
# Ldap server host (specify multiple hosts space separated)
host = "ldap_1.cisco.com"
# Default port is 389 or 636 if use_ssl = true
port = 10648
# Set to true if ldap server supports TLS
use_ssl = true
# set to true if you want to skip ssl cert validation
ssl_skip_verify = true
# set to the path to your root CA certificate or leave unset to use system defaults
#root_ca_cert = "/etc/openldap/certs/ldap_local.cer"

# Search user bind dn
bind_dn = "uid=admin,ou=system"
# Search user bind password
bind_password = 'secret'

# User search filter, for example "(cn=%s)" or "(sAMAccountName=%s)" or "(uid=%s)"
search_filter = "(uid=%s)"

# An array of base dns to search through
search_base_dns = ["ou=users,dc=sprint,dc=com"]
#search_base_dns = ["ou=groups,dc=sprint,dc=com"]

# In POSIX LDAP schemas, without memberOf attribute a secondary query must be made for
groups.
# This is done by enabling group_search_filter below. You must also set member_of= "cn"
# in [servers.attributes] below.
# Users with nested/recursive group membership and an LDAP server that supports
LDAP_MATCHING_RULE_IN_CHAIN
# can set group_search_filter, group_search_filter_user_attribute, group_search_base_dns
and member_of
# below in such a way that the user's recursive group membership is considered.
#
# Nested Groups + Active Directory (AD) Example:
#
# AD groups store the Distinguished Names (DNs) of members, so your filter must
# recursively search your groups for the authenticating user's DN. For example:
#
# group_search_filter = "(member:1.2.840.113556.1.4.1941:=%s)"
# group_search_filter_user_attribute = "distinguishedName"
# group_search_base_dns = ["ou=groups,dc=grafana,dc=org"]
#
# [servers.attributes]
# ...
# member_of = "distinguishedName"

## Group search filter, to retrieve the groups of which the user is a member (only set if
memberOf attribute is not available)
#group_search_filter = "(cn=%s)"
#group_search_filter = "(&(objectClass=*)(cn=%s))"
## Group search filter user attribute defines what user attribute gets substituted for %s
in group_search_filter.
## Defaults to the value of username in [server.attributes]
## Valid options are any of your values in [servers.attributes]
## If you are using nested groups you probably want to set this and member_of in
## [servers.attributes] to "distinguishedName"
group_search_filter_user_attribute = "cn"
## An array of the base DNs to search through for groups. Typically uses ou=groups
group_search_base_dns = ["ou=groups,dc=sprint,dc=com"]
#group_search_base_dns = ["cn=Roles,ou=groups,dc=sprint,dc=com"]

# Specify names of the ldap attributes your ldap uses
[servers.attributes]
```

```
name = "cn"
surname = "sn"
username = "uid"
member_of = "cn"
email = "email"

# Map ldap groups to grafana org roles
[[servers.group_mappings]]
group_dn = "cn=Admin,ou=groups,dc=sprint,dc=com"
org_role = "Admin"
# The Grafana organization database id, optional, if left out the default org (id 1) will
be used
# org_id = 1

[[servers.group_mappings]]
group_dn = "cn=User,ou=groups,dc=sprint,dc=com"
org_role = "Editor"

#[[servers.group_mappings]]
# If you want to match all (or no ldap groups) then you can use wildcard
#group_dn = "*"
#org_role = "Viewer"
```

Step 5 Restart Grafana server by running the following command:

```
service grafana-server restart
```

Step 6 Log in to Grafana using LDAP user credentials.

Mongo Database

Purpose

MongoDB is used to manage session storage efficiently and address key requirements: Low latency reads/writes, high availability, multi-key access and so on.

CPS support different models of mongo database based on CPS deployment like AIO, HA or Geo-redundancy. Not all of the databases listed below may be used in your CPS deployment.

To rotate the mongoDB logs on the Session Manager VM, open the mongoDB file by executing the following command:

cat /etc/logrotate.d/mongodb

You will have output as similar to the following:

```
{
daily
rotate 5
copytruncate
create 640 root root
sharedscripts
postrotate
endscript
}
```

In the above script the mongoDB logs are rotated daily and it ensures that it keeps the latest 5 backups of these log files.

HA

The standard definition for supported replica-set defined in configuration file. This configuration file is self-explanatory which contains replica-set, set-name, hostname, port number, data file path and so on.

Location: `/etc/broadhop/mongoConfig.cfg`

Table 4: HA Mongo Databases

Database Name	Port Number	Primary DB Host	Secondary DB Host	Arbiter	Purpose
session_cache	27717	sessionmgr01	sessionmgr02	pcrfclient01	Session database
balance_mgmt	27718	sessionmgr01	sessionmgr02	pcrfclient01	Quota/Balance database
audit	27725	sessionmgr01	sessionmgr02	pcrfclient01	Reporting database
spr	27720	sessionmgr01	sessionmgr02	pcrfclient01	USuM database
cust_ref_data	27717	sessionmgr01	sessionmgr02	pcrfclient01	Custom Reference Data

Note The port number configuration is based on what is configured in each of the respective Policy Builder plug-ins. Refer to the *Plug-in Configuration* chapter of the *CPS Mobile Configuration Guide* for correct port number and ports defined in mongo configuration file.

AIO

The All-in-One deployment mongo database runs on ports 27017 and 27729.

Table 5: AIO Mongo Databases

Database Name	Port Number	Purpose
All	27017	This port is used for all the databases.

Important While choosing mongo ports for replica-sets, consider the following:

- Port is not in use by any other application. To check it, login to VM on which replica-set is to be created and execute the following command:

  ```
  netstat -lnp | grep <port_no>
  ```

  If no process is using same port then port can be chosen for replica-set for binding.

- Port number used should be greater than 1024 and not in ephemeral port range i.e, not in between following range :

  net.ipv4.ip_local_port_range = 32768 to 61000

CLI Access

Use the following commands to access the mongoDB CLI:

HA:

Login to pcrfclient01 or pcrfclient02 and run: `diagnostics.sh --get_replica_status`

This command will output information about the databases configured in the CPS cluster.

Note If a member is shown in an unknown state, it is likely that the member is not accessible from one of other members, mostly an arbiter. In that case, you must go to that member and check its connectivity with other members.

Also, you can login to mongo on that member and check its actual status.

AIO:

```
mongo --port 27017
```

Protocol

Not applicable.

Port

Not applicable.

Accounts and Roles

Restrict MongoDB Access for Readonly Users: If firewall is enabled on system, then on all VMs for all readonly users, IP table rule will be created for outgoing connections to reject outgoing traffic to MongoDB replica sets.

For example, rule similar to the following is created.

```
REJECT tcp -- anywhere sessionmgr01 tcp dpt:27718 owner GID match qns-ro reject-with
icmp-port-unreachable
```

With this, qns-ro user has restricted MongoDB access on sessionmgr01 on port 27718. Such rules are added for all readonly users who are part of qns-ro group for all replica sets.

OSGi Console

Purpose

CPS is based on Open Service Gateway initiative (OSGi) and OSGi console is a command-line shell which can be used for analyzing problems at OSGi layer of the application.

CLI Access

Use the following command to access the OSGi console:

`telnet` <*ip*> <*port*>

The following commands can be executed on the OSGi console:

`ss` : List installed bundle status.

`start` <*bundle-id*> : Start the bundle.

`stop` <*bundle-id*> : Stop the bundle.

`diag` <*bundle-id*> : Diagnose the bundle.

Sharding Commands

Use the following OSGi commands to add or remove shards:

Table 6: Sharding Commands

Command	Description
`listshards`	Lists all the shards.
`removeshard <shard id>`	Marks the shard for removal. If shard is non-backup, rebalance is required for shard to be removed fully. If shard is backup, it does not require rebalance of sessions and hence would be removed immediately.
`rebalance <rate limit>`	Rebalances the buckets and migrates session with rate limit. Rate limit is optional. If rate limit is passed, it is applied at rebalance.
`rebalancebg <rate limit>`	Rebalances the buckets and schedules background task to migrate sessions. Rate limit is optional. If rate limit is passed, it is applied at rebalance.
`rebalancestatus`	Displays the current rebalance status. Status can be one of the following: • Rebalance is running (Remaining buckets: <pending count>) • Rebalance is required • Rebalanced
`rebuildAllSkRings`	In order for CPS to identify a stale session from the latest session, the secondary key mapping for each site stores the primary key in addition to the bucket ID and the site ID, that is, Secondary Key = <Bucket Id>; <Site Id>; <Primary Key>. To enable this feature, add the flag `-Dcache.config.version=1` in the `/etc/broadhop/qns.conf` file. Enabling this flag and running `rebuildAllSkRings` starts the data migration for the new version so that CPS can load the latest version of the session.
`skRingRebuildStatus`	Displays the status of the migration and the current cache version.

CPS Alarm Commands

Use the following OSGi command to get the information related to open application alarms in CPS:

Table 7: Alarm Commands

Command	Description
`listalarms`	To list the open/active application alarms since last restart of policy server (QNS) process on pcrfclient01/02 VM.

Example:

```
osgi> listalarms
Active Application Alarms
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,10:47:34,
051+0000  msg="3001:Host: site-host-gx Realm: site-gx-client.com is down"
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,10:47:34,
048+0000  msg="3001:Host: site-host-sd Realm: site-sd-client.com is down"
id=1000 sub_id=3001  event_host=lb01  status=down date=2017-11-22,10:45:17,
927+0000  msg="3001:Host: site-server Realm: site-server.com is down"
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,10:47:34,
091+0000  msg="3001:Host: site-host-rx Realm: site-rx-client.com is down"
id=1000 sub_id=3002  event_host=lb02  status=down date=2017-11-22,10:47:34,
111+0000  msg="3002:Realm: site-server.com:applicationId: 7:all peers are down"
```

Ports

pcrfclientXX:

- Control Center: 9091
- Policy Builder: 9092

lbXX:

- iomanager: 9091
- Diameter Endpoints: 9092, 9093, 9094 ...

qnsXX: 9091

Ports should be blocked using a firewall to prevent access from outside the CPS cluster.

Accounts and Roles

Not applicable.

Policy Builder GUI

Purpose

Policy Builder is the web-based client interface for the configuration of policies in Cisco Policy Suite.

URL and Port

HA: https://<*lbvip01*>:7443/pb

AIO: http://<*ip*>:7070/pb

Protocol

HTTPS/HTTP

Accounts and Roles

Initial accounts are created during the software installation. Refer to the *CPS Operations Guide* for commands to add users and change passwords.

REST API

Purpose

To allow initial investigation into a Proof of Concept API for managing a CPS System and Custom Reference Data related through an HTTPS accessible JSON API.

CLI Access

This is an HTTPS/Web interface and has no Command Line Interface.

URL and Port

API: http://<*Cluster Manager IP*>:8458

Documentation: http://<*Cluster Manager IP*>:7070/doc/index.html

Accounts and Roles

Initial accounts are created during the software installation. Refer to the *CPS Operations Guide* for commands to add users and change passwords.

Rsyslog

Purpose

Enhanced log processing is provided using Rsyslog.

Rsyslog logs Operating System (OS) data locally (`/var/log/messages` etc.) using the `/etc/rsyslog.conf` and `/etc/rsyslog.d/*conf` configuration files.

rsyslog outputs all WARN level logs on CPS VMs to `/var/log/warn.log` file.

On all nodes, Rsyslog forwards the OS system log data to lbvip02 via UDP over the port defined in the logback_syslog_daemon_port variable as set in the CPS deployment template (Excel spreadsheet). To download the most current CPS Deployment Template (`/var/qps/install/current/scripts/deployer/templates/QPS_deployment_config_template.xlsm`), refer to the *CPS Installation Guide for VMware* or *CPS Release Notes* for this release.

Additional information is available in the Logging chapter of the *CPS Troubleshooting Guide*. Refer also to http://www.rsyslog.com/doc/ for the Rsyslog documentation.

CLI Access

Not applicable.

Protocol

UDP

Port

6514

Accounts and Roles

Account and role management is not applicable.

Rsyslog Customization

CPS provides the ability to configure forwarding of consolidated syslogs from rsyslog-proxy on Policy Director VMs to remote syslog servers (refer to *CPS Installation Guide for VMware*). However, if additional customizations are made to rsyslog configuration to forward logs to external syslog servers in customer's network for monitoring purposes, such forwarding must be performed via dedicated action queues in rsyslog. In the absence of dedicated action queues, when rsyslog is unable to deliver a message to the remote server, its main message queue can fill up which can lead to severe issues, such as, preventing SSH logging, which in turn can prevent SSH access to the VM.

Sample configuration for dedicated action queues is available in the *Logging* chapter of the *CPS Troubleshooting Guide*. Refer to rsyslog documentation on http://www.rsyslog.com/doc/v5-stable/concepts/queues.html for more details about action queues.

SVN Interface

Apache™ Subversion (SVN) is the versioning and revision control system used within CPS. It maintains all the CPS policy configurations and has repositories in which files can be created, updated and deleted. SVN maintains the file difference each time any change is made to a file on the server and for each change it generates a revision number.

In general, most interactions with SVN are performed via Policy Builder.

CLI Access

Use the following commands to access SVN:

Get all files from the server:

```
svn checkout --username <username> --password <password> <SVN Repository URL> <Local Path>
```

Example:

```
svn checkout --username broadhop --password broadhop
http://pcrfclient01/repos/configuration/root/configuration
```

If <*Local Path*> is not provided, files are checked out to the current directory.

Store/check-in the changed files to the server:

```
svn commit --username <username> --password <password> <Local Path> -m "modified config"
```

Example:

```
svn commit --username broadhop --password broadhop /root/configuration -m "modified config"
```

Update local copy to latest from SVN:

```
svn update <Local Path>
```

Example:

```
svn update /root/configuration/
```

Check current revision of files:

```
svn info <Local Path>
```

Example:

```
svn info /root/configuration/
```

Note Use `svn --help` for a list of other commands.

Protocol

HTTP

Port

80

Accounts and Roles

CPS 7.0 and Higher Releases

Add User with Read Only Permission

From the pcrfclient01 VM, run **adduser.sh** to create a new user.

```
/var/qps/bin/support/adduser.sh
```

Note This command can also be run from the Cluster Manager VM, but you must include the OAM (PCRFCLIENT) option:

```
/var/qps/bin/support/adduser.sh pcrfclient
```

Example:

```
[root@pcrfclient01 /]# /var/qps/bin/support/adduser.sh
Enter username: <username>
Enter group for the user: <any group>
Enter password:
Re-enter password:
```

Add User with Read/Write Permission

By default, the **adduser.sh** script creates a new user with read-only permissions. For read-write permission, you must assign the user to the **qns-svn** group and then run the **vm-init** command.

From the pcrfclient01 VM, run the **adduser.sh** script to create the new user.

Run the following command on both pcrfclient01 and pcrfclient02 VMs:

```
/etc/init.d/vm-init
```

You can now login and commit changes as the newly created user.

Change Password

From the pcrfclient01 VM, run the **change_passwd.sh** script to change the password of a user.

```
/var/qps/bin/support/change_passwd.sh
```

Example:

```
[root@pcrfclient01 /]# /var/qps/bin/support/change_passwd.sh
Enter username whose password needs to be changed: user1
Enter new password:
Re-enter new password:
```

CPS Versions Earlier than 7.0

Perform all of the following commands on both the pcrfclient01 and pcrfclient02 VMs.

Add User

Use the htpasswd utility to add a new user

```
htpasswd -mb /var/www/svn/.htpasswd <username> <password>
```

Example:

```
htpasswd -mb /var/www/svn/.htpasswd user1 password
```

In some versions, the password file is `/var/www/svn/password`

Provide Access

Update the user role file /var/www/svn/users-access-file and add the username under `admins` (for read/writer permissions) or `nonadmins` (for read-only permissions). For example:

```
[groups]
admins = broadhop
nonadmins = read-only, user1
[/]
@admins = rw
@nonadmins = r
```

Change Password

Use the htpasswd utility to change passwords.

```
htpasswd -mb /var/www/svn/.htpasswd <username> <password>
```

Example:

```
htpasswd -mb /var/www/svn/.htpasswd user1 password
```

TACACS+ Interface

Purpose

CPS 7.0 and above has been designed to leverage the Terminal Access Controller Access Control System Plus (TACACS+) to facilitate centralized management of users. Leveraging TACACS+, the system is able to provide system-wide authentication, authorization, and accounting (AAA) for the CPS system.

Further the system allows users to gain different entitlements based on user role. These can be centrally managed based on the attribute-value pairs (AVP) returned on TACACS+ authorization queries.

CLI Access

No CLI is provided.

Port

CPS communicates to the AAA backend using IP address/port combinations configured by the operator.

Account Management

Configuration is managed by the Cluster Management VM which deploys the `/etc/tacplus.conf` and various PAM configuration files to the application VMs. For more account management information, refer to TACACS+ Service Requirements, on page 66.

For more information about TACACS+, refer to the following links:

- TACAC+ Protocol Draft: http://tools.ietf.org/html/draft-grant-tacacs-02
- Portions of the solution reuse software from the open source pam_tacplus project hosted at: https://github.com/jeroennijhof/pam_tacplus

For information on CLI commands, refer to Accessing the CPS CLI, on page 52.

Unified API

Purpose

Unified APIs are used to reference customer data table values.

URL and Port

HA: https://<*lbvip01*>:8443/ua/soap

AIO: http://<*ip*>:8080/ua/soap

Protocol

HTTPS/HTTP

Accounts and Roles

Currently there is no authorization for this API

Accessing the CPS CLI

sudo supports a plugin architecture for security policies and input/output logging. The default security policy is *sudoers*, which is configured via the file `/etc/sudoers`, contains the rules that users must follow when using the sudo command.

sudo allows a system administrator to delegate authority to give certain users (or groups of users) the ability to run some (or all) commands as root or another user while providing an audit trail of the commands and their arguments.

For example: `%adm ALL=(ALL) NOPASSWD: ALL`

This means that any user in the administrator group on any host may run any command as any user without a password. The first ALL refers to hosts, the second to target users, and the last to allowed commands.

When an authenticated user has one of the above group permissions, they can access the CPS CLI and run predefined commands available to that user role. A list of commands available after authentication can be viewed using the `sudo -l` command (-l for list), or any user with root privileges can use `sudo -l -U <qns-role>` to see the available command for a specific Policy Server (qns) role.

The `/etc/sudoers` file contains user specifications that define the commands that users may execute. When sudo is invoked, these specifications are checked in order, and the last match is used. A user specification looks like this at its most basic:

```
User Host = (Runas) Command
```

Read this as "User may run Command as the Runas user on Host". Any or all of the above may be the special keyword ALL, which always matches. User and Runas may be usernames, group names prefixed with %, numeric UIDs prefixed with #, or numeric GIDs prefixed with %#. Host may be a hostname, IP address, or a whole network (for example, 192.0.2.0/24), but not 127.0.0.1.

Group Identifiers

gid

The group identifier of the TACACS+ authenticated user on the VM nodes. This value should reflect the role assigned to a given user, based on the following values:

- group id=500 (qns)

  The group identifier used by Policy Server (qns) user in application.

- group id=501 (qns-su)

  This group identifier should be used for users that are entitled to attain superuser (or 'root') access on the CPS VM nodes.

- group id=504 (qns-admin)

  This group identifier should be used for users that are entitled to perform administrative maintenance on the CPS VM nodes.

Note To execute administrative scripts from qns-admin, prefix the command with `sudo`. For example

```
sudo stopall.sh
```

- group id=505 (qns-ro)

  This group identifier should be used for users that are entitled to read-only access to the CPS VM nodes.

When an authenticated user has one of the above group permissions, they can access the CPS CLI and run predefined commands available to that user role. A list of commands available after authentication can be viewed using the `sudo -l` command (-l for list), or any user with root privileges can use `sudo -l -U <qns-role>` to see the available command for a specific Policy Server (qns) role.

For more information, refer to https://www.sudo.ws/intro.html.

home

The user's home directory on the CPS VM nodes. To enable simpler management of these systems, the users should be configured with a pre-deployed shared home directory based on the role they are assigned with the gid.

- home=/home/qns-su should be used for users in the 'qns-su' group (gid=501)
- home=/home/qns-admin should be used for users in the 'qnsadmin' group (gid=504)
- home=/home/qns-ro should be used for users in the 'qns-ro' group (gid=505)

Multi-user Policy Builder

Multiple users can be logged into Policy Builder at the same time.

In the event that two users attempt to make changes on same screen and one user saves their changes to the client repository, the other user may receive errors. In such cases the user must return to the login page, revert the configuration, and repeat their changes.

This section covers the following topics:

- Create Users, on page 53
- Revert Configuration, on page 54

Create Users

Step 1 Log in to the Cluster Manager.

Step 2 Add a user to CPS by executing:

```
adduser.sh
```

Step 3 When prompted for the user’s group, set ‘qns-svn’ for read-write permissions or ‘qns-ro’ for read-only permissions.

- To check if a user already exists, login in as root and enter `su username`.
- To check a user's 'groups', enter `groups username`.
- To change a user's password, use the `change_passwd.sh` command.

Refer to CPS Commands, on page 183 for more information about these commands.

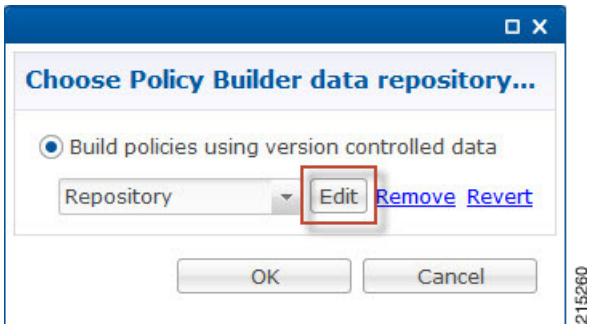
Revert Configuration

The user can revert the configuration if changes since the last publish/save to client repository are not wanted.

This can also be necessary in the case of a 'syn conflict' error where both pcrfclient01 and pcrfclient02 are in use at the same time by different users and publish/save to client repository changes to the same file. The effect of reverting changes is that all changes since the publish/save to client repository will be undone.

Step 1 On the Policy Builder login screen, verify the user for which changes need to be reverted is correct. This can be done by clicking **Edit** and verifying that the Username and Password fields are correct.

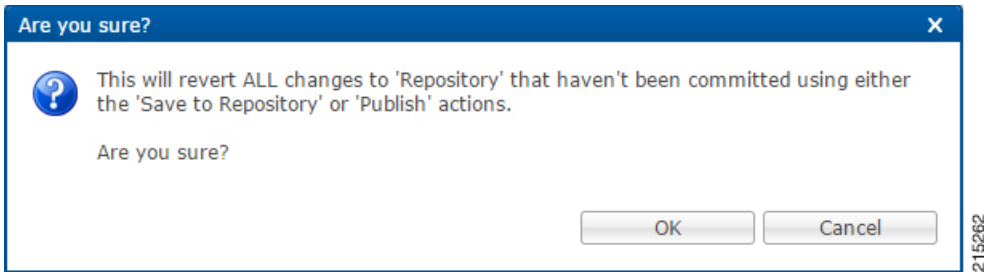
Figure 2: Verifying the User



Step 2 Click **Revert**.

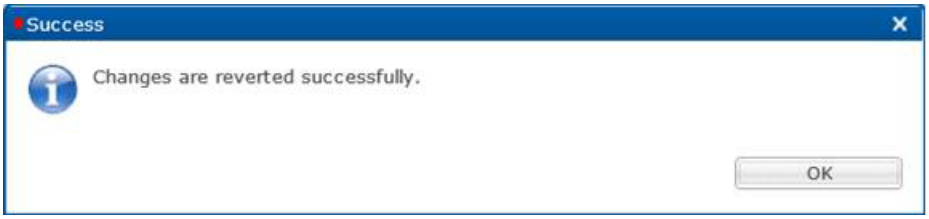
The following confirmation dialog opens.

Figure 3: Revert Confirmation Message



Step 3 Click **OK** to revert back to the earlier configuration. The following dialog confirms that the changes are reverted successfully.

Figure 4: Success Confirmation Message



Publishing Data

This section describes publishing Cisco Policy Builder data to the Cisco Policy Server. Publishing data occurs in the Cisco Policy Builder client interface, but affects the Cisco Policy Server. Refer to the *CPS Mobile Configuration Guide* for steps to publish data to the server.

Cisco Policy Builder manages data stored in two areas:

- The Client Repository stores data captured from the Policy Builder GUI in Subversion. This is a place where trial configurations can be developed and saved without affecting the operation of the Cisco Policy Builder server data.

  The default URL is http://pcrfclient01/repos/configuration.

- The Server Repository is where a copy of the client repository is created/updated and where the CPS picks up changes. This is done on Publish from Policy Builder.

  Note Publishing will also do a Save to Client Repository to ensure the Policy Builder and Server configurations are not out of sync.

  The default URL is http://pcrfclient01/repos/run.

Control Center Access

After the installation is complete, you need to configure the Control Center access. This is designed to give the customer a customized Control Center username.

Add a Control Center User

Step 1 Login to the Cluster Manager VM.

Step 2 Execute the following script to add a Control Center user.

```
/var/qps/bin/support/adduser.sh
```

Note To add a user with 'read/write' access to Control Center, their group should be 'qns'. To add a user with 'read' access to Control Center, their group should be 'qns-ro'.

Example:

```
/var/qps/bin/support/adduser.sh
Enter username: username
Enter group for the user: groupname
Enter password: password
Re-enter password: password
```

This example adds *username* to all the VMs in the cluster.

Update Control Center Mapping

This section describes updating Control Center mapping of read-write/read-only to user groups (Default: qns and qns-ro respectively).

Step 1 Login to the Cluster Manager VM.

Step 2 Update `/etc/broadhop/authentication-provider.xml` to include the group mapping for the group you want to use.

Note Make sure that this group exists on at least the Policy Server (QNS) VMs or adding users will fail due to no group available (there should be an entry in `/etc/group`).

In the following example, the 'test' group has been added as an read-write mapping for Control Center - updated line in bold:

```
<beans:beans xmlns="http://www.springframework.org/schema/security"
  xmlns:beans="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
classpath:/org/springframework/beans/factory/xml/spring-beans-3.0.xsd
  http://www.springframework.org/schema/security
classpath:/org/springframework/security/config/spring-security-3.0.xsd">
<beans:bean id="authenticationProvider"
class="com.broadhop.ui.security.server.pam.PamAuthenticationProvider">
    <!-- change the key value to be the customer's role that maps to the cisco role. -->
    <beans:property name="roleMap">
      <beans:map>
        <beans:entry key="qns" value="ROLE_SUMADMIN" />
        <beans:entry key="test" value="ROLE_SUMADMIN" />
        <beans:entry key="qns-ro" value="ROLE_READONLY" />
      </beans:map>
    </beans:property>
  </beans:bean>

  <authentication-manager>
    <authentication-provider ref="authenticationProvider" />
  </authentication-manager>

</beans:beans>
```

Step 3 Run `syncconfig.sh` to put this file on all VMs.

Step 4 Restart the CPS system, so that the changes done above are reflected in the VMs:

```
restartall.sh
```

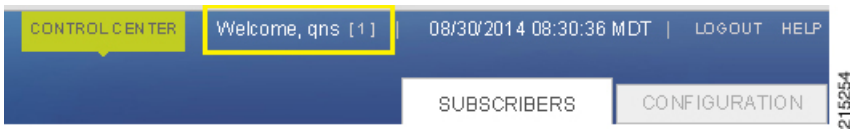
To add a new user to Control Center and specify the group you have specified in the configuration file above, refer to Add a Control Center User, on page 55.

Multiple Concurrent User Sessions

CPS Control Center supports session limits per user. If the user exceeds the configured session limit, they are not allowed to log in. CPS also provides notifications to the user when other users are already logged in.

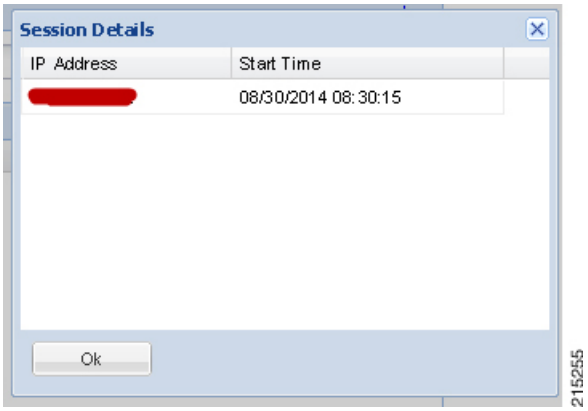
When a user logs in to Control Center, a Welcome message displays at the top of the screen. A session counter is shown next to the username. This represents the number of login sessions for this user. In the following example, this user is logged in only once ([1]).

Figure 5: Welcome Message



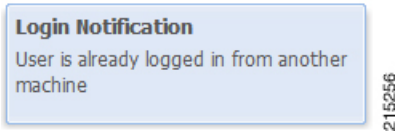
The user can click the session counter ([1]) link to view details for the session(s), as shown below.

Figure 6: Viewing Session Details



When another user is already logged in with the same username, a notification displays for the second user in the bottom right corner of the screen, as shown below.

Figure 7: Login Notification for a Second User



The first user also receives a notification, as shown, and the session counter is updated to [2].

Figure 8: Login Notification for First User

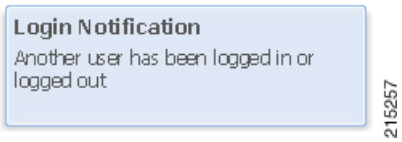
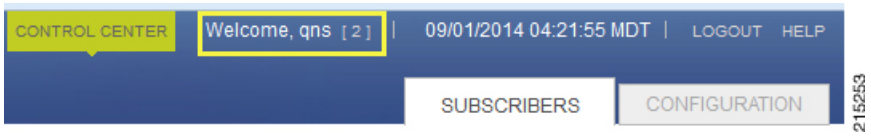


Figure 9: Indication of Two Users with Same Username



These notifications are not displayed in real time; CPS updates this status every 30 seconds.

Configure Session Limit

The session limit can be configured by the runtime argument, which can be configured in the qns.conf file.

-Dcc.user.session.limit=3 (default value is 5)

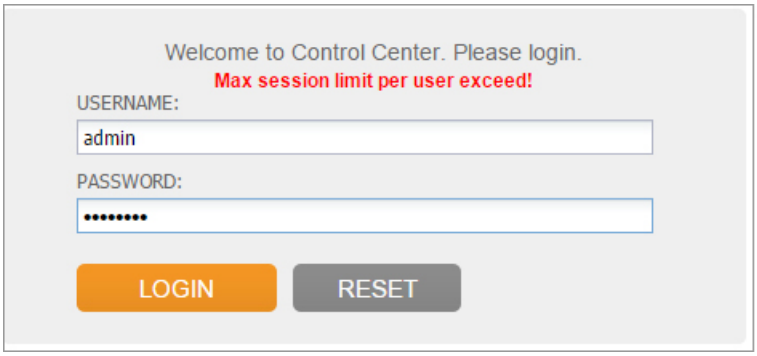
Configure Session Timeout

The default session timeout can be changed by editing the following file on the Policy Server (QNS) instance:

```
./opt/broadhop/qns-1/plugins/com.broadhop.ui_3.5.0.release/war/WEB-INF/web.xml

<!-- timeout after 15 mins of inactivity -->
<session-config>
<session-timeout>15</session-timeout>
    <cookie-config>
<http-only>true</http-only>
</cookie-config>
</session-config>
```

Note The same timeout value must be entered on all Policy Server (QNS) instances.

When the number of sessions of the user exceeds the session limit, the user is not allowed to log in and receives the message "`Max session limit per user exceed!`"



Important Notes

If a user does not log out and then closes their browser, the session remains alive on the server until the session times out. When the session timeout occurs, the session is deleted from the memcached server. The default session timeout is 15 minutes. This is the idle time after which the session is automatically deleted.

When a Policy Server (QNS) instance is restarted, all user/session details are cleared.

When the memcached server is restarted without also restarting the Policy Server (QNS) instance, all http sessions on the Policy Server (QNS) instance are invalidated. In this case the user is asked to log in again and after that, the new session is created.

Enabling Authentication and Authorization for CRD API

Update the HAProxy configuration to enable authentication and authorization mechanism in the CRD API module.

There are two options to include a username and password in an API request:

1. Include the username and password directly in the request as shown:

   https://<username>:<password>@<lbvip02>:8443/custrefdata/_checksum

2. Add an authentication header to the request as shown:

   Authorization: Basic <base64 encoded value of username:password>

Step 1 Back up the `/etc/haproxy/haproxy.cfg` file before making modifications in the following steps.

Step 2 Edit `/etc/haproxy/haproxy.cfg` on lb01/lb02 and add a userlist with at least one username and password.

Use the following syntax:

```
  userlist <userlist name>
 user <username1> password <encrypted password>
```

For example:

```
userlist cps_user_list
user readonly password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
user apiuser password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
```

Run the following command to generate an encrypted password:

```
/sbin/grub-crypt --sha-512
```

For example:

```
[root@host ~]# /sbin/grub-crypt --sha-512
Password:
Retype password:
<encrypted password output>
```

Step 3 Add the following line in frontend `https-api` to enable Authentication and Authorization for CRD REST API and create a new backend server as `crd_api_servers` to intercept CRD REST API requests:

```
mode http
 acl crd_api path_beg -i  /custrefdata/
 use_backend crd_api_servers if crd_api

 backend crd_api_servers
         mode http
         balance roundrobin
         option httpclose
        option abortonclose
        server qns01_A qns01:8080 check inter 30s
        server qns02_A qns02:8080 check inter 30s
```

Step 4 Update frontend `https_all_servers` by replacing `api_servers` with `crd_api_servers` for CRD API as follows:

```
acl crd_api path_beg -i /custrefdata/

use_backend crd_api_servers if crd_api
```

Step 5 To enable the authentication, edit `/etc/haproxy/haproxy.cfg` on lb01/lb02 and add the following lines in the `backend crd_api_servers`:

```
 acl validateAuth http_auth(<userlist_name>)
 http-request auth unless validateAuth
```

Map the userlist created in *Step 2* with the acl as follows:

```
acl validateAuth http_auth(<userlist name>)
```

Step 6 To enable the authorization, add at least one group with the user in `userlist` created in *Step 2* as follows:

```
 group qns-ro users readonly
```

For example:

```
userlist cps_user_list
group qns-ro users readonly
user readonly password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
user apiuser password
$6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrmeAUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
```

Step 7 Add the following in the `backend crd_api_servers` to set read-only permission (GET HTTP operation) for group of users:

```
 acl authoriseUsers http_auth_group(<user-list-name>) <group-name>

 http-request deny if !METH_GET authoriseUsers
```

Map the group created in *Step 6* with the acl in the following line:

```
 acl authorizeUsers http_auth_group(<userlist name>) <group-name>
```

Example:

HAProxy Configuration Example

```
userlist cps_user_list
      group qns-ro users readonly

      user readonly password $6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrme
AUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1
```

```
      user apiuser password $6$xRtThhVpS0w4lOoS$pyEM6VYpVaUAxO0Pjb61Z5eZrme
AUUdCMF7D75BXKbs4dhNCbXjgChVE0ckfLDp4T2CsUzzNkoqLRdn7RbAAU1

frontend https-api
description API
bind lbvip01:8443 ssl crt /etc/ssl/certs/quantum.pem

mode http
acl crd_api path_beg -i  /custrefdata/
use_backend crd_api_servers if crd_api
default_backend api_servers
reqadd X-Forwarded-Proto:\ https if { ssl_fc }
frontend https_all_servers
description Unified API,CC,PB,Grafana,CRD-API,PB-API
bind lbvip01:443 ssl crt /etc/ssl/certs/quantum.pem no-sslv3 no-tlsv10
ciphers ECDH+AESGCM:DH+AESGCM:ECDH+AES256:DH+AES256:ECDH+AES128:DH+AES:RSA+AESGCM:RSA+AES:!
aNULL:!eNULL:!LOW:!     3DES:!MD5:!EXP:!PSK:!SRP:!DSS
mode http
acl crd_api path_beg -i  /custrefdata/
use_backend crd_api_servers if crd_api
backend crd_api_servers
       mode http
       balance roundrobin
       option httpclose
       option abortonclose
       server qns01_A qns01:8080 check inter 30s
       server qns02_A qns02:8080 check inter 30s
       acl validateAuth http_auth(cps_user_list)
       acl authoriseUsers http_auth_group(cps_user_list) qns-ro
       http-request auth unless validateAuth
       http-request deny if !METH_GET authoriseUsers
```

Note The haproxy.cfg file is generated by the Puppet tool. Any manual changes to the file in lb01/lb02 would be reverted if the pupdate or vm-init scripts are run.

Unified API Security: Access Privileges

By default, the CPS Unified API does not require username and password authentication. To enable authentication, refer to Enable Authentication for Unified API, on page 62.

There are two options to include a username and password in an API request:

- Include the username and password directly in the request. For example:

  https://<username>:<password>@<lbvip02>:8443/ua/soap

- Add an authentication header to the request:

  `Authorization: Basic <base64 encoded value of username:password>`

  For example:

  ```
  wget -d -O - --header="Authorization: Basic cG9ydGFXNjbzEyMwo="
  https://lbvip02:8443/ua/soap/keepalive
  ```

Enable Authentication for Unified API

HAProxy is used to secure and balance calls to the CPS Unified API.

Step 1 Back up the `/etc/haproxy/haproxy.cfg` file before making modifications in the following steps.

Step 2 Edit `/etc/haproxy/haproxy.cfg` on lb01/lb02 and add a userlist with at least one username and password.

Use the following syntax:

```
userlist <userlist name>
user <username1> password <encrypted password>
user <username2> insecure-password <plain text password>
```

For example:

```
userlist L1
user apiuser password $6$eC8mFOWMcRnQo7FQ$C053tv5T2mPlmGAta0ukH87MpK9aLPtWgCEK
```

Step 3 Run the following command to generate an encrypted password:

```
/sbin/grub-crypt --sha-512
```

For example:

```
[root@host ~]# /sbin/grub-crypt --sha-512
Password:
Retype password:
<encrypted password output>
```

Step 4 Edit `/etc/haproxy/haproxy.cfg` on lb01/lb02 to configure HAProxy to require authentication. Add the following 4 lines to the haproxy.cfg file:

```
acl validateAuth http_auth(<userlist_name>)
acl unifiedAPI path_beg -i /ua/soap
http-request allow if !unifiedAPI
http-request auth unless validateAuth
```

The userlist created in *Step 2* needs to be mapped with the acl in the following line:

```
acl validateAuth http_auth(<userlist name>)
```

For example:

```
frontend https-api
description Unified API
bind lbvip01:8443 ssl crt /etc/ssl/certs/quantum.pem
default_backend api_servers
reqadd X-Forwarded-Proto:\ https if { ssl_fc }
backend api_servers
mode http
balance roundrobin
option httpclose
option abortonclose
option httpchk GET /ua/soap/keepalive
server qns01_A qns01:8080 check inter 30s
server qns02_A qns02:8080 check inter 30s
server qns03_A qns03:8080 check inter 30s
server qns04_A qns04:8080 check inter 30s
acl validateAuth http_auth(L1)
acl unifiedAPI path_beg -i /ua/soap
```

```
http-request allow if !unifiedAPI
http-request auth unless validateAuth
```

The configuration above applies authentication on context /ua/soap, which is the URL path of the Unified API.

Note The haproxy.cfg file is generated by the Puppet tool. Any manual changes to the file in lb01/lb02 would be reverted if the pupdate or vm-init scripts are run.

WSDL and Schema Documentation

In order to access the Unified API WSDL while using authentication change the following line:

`acl unifiedAPI path_beg -i /ua/soap`

to

`acl unifiedAPI path_beg -i /ua/.`

The default address for the WSDL is `https://<lbvip01>:8443/ua/wsdl/UnifiedApi.wsdl`

The Unified API contains full documentation in an html format that is compatible with all major browsers.

The default address is `https://<HA-server-IP>:8443/ua/wsdl/UnifiedApi.xsd`

Note Run the **about.sh** command from the Cluster Manager to display the actual addresses as configured in your deployment.

Enabling Unified API Access on HTTP Port 8080

CPS 7.x onward uses HTTPS on port 8443 for Unified API access. To enable HTTP support (like pre-7.0) on port 8080, perform the following steps:

Note Make sure to open port 8080 if firewall is used on the setup.

Step 1 Create the following directories (ignore File exists error), on Cluster Manager:

```
/bin/mkdir -p /var/qps/env_config/modules/custom/templates/etc/haproxy
/bin/mkdir -p /var/qps/env_config/modules/custom/templates/etc/monit.d
/bin/mkdir -p /var/qps/env_config/nodes
```

Step 2 Create the file `/var/qps/env_config/modules/custom/templates/etc/haproxy/haproxy-soaphttp.erb` with the following contents on Cluster Manager:

- Change XXXX with the Unified API interface hostname or IP
- In this example, we are adding 10 Policy Servers (QNS). You can add/remove the number of Policy Servers (QNS) depending on your network requirements.

```
global
  daemon
  nbproc       1          # number of processing cores
  stats socket /tmp/haproxy-soaphttp
defaults
  timeout client    60000ms       # maximum inactivity time on the client side
  timeout server    180000ms      # maximum inactivity time on the server side
  timeout connect   60000ms       # maximum time to wait for a connection attempt to a server to
succeed

  log             127.0.0.1       local1 err

listen pcrf_proxy XXXX:8080 ----------- > where, XXXX, is Unified API interface hostname or IP
        mode http
        balance roundrobin
        option httpclose
        option abortonclose
        option httpchk GET /ua/soap/KeepAlive
        server qns01_A qns01:8080 check inter 30s
        server qns02_A qns02:8080 check inter 30s
        server qns03_A qns03:8080 check inter 30s
        server qns04_A qns04:8080 check inter 30s
        server qns05_A qns05:8080 check inter 30s
        server qns06_A qns06:8080 check inter 30s
        server qns07_A qns07:8080 check inter 30s
        server qns08_A qns08:8080 check inter 30s
        server qns09_A qns09:8080 check inter 30s
        server qns10_A qns10:8080 check inter 30s
```

Step 3 Create the file `/var/qps/env_config/modules/custom/templates/etc/monit.d/haproxy-soaphttp` with the following contents on Cluster Manager:

```
check process haproxy-soaphttp with pidfile /var/run/haproxy-soaphttp.pid
start = "/usr/bin/systemctl start haproxy-soaphttp"
stop = "/usr/bin/systemctl stop haproxy-soaphttp"
```

Step 4 Create or modify the `/var/qps/env_config/nodes/lb.yaml` file with the following contents on Cluster Manager:

If the file exists then just add `custom::soap_http:`

```
classes:
  qps::roles::lb:
  custom::soap_http:
```

Step 5 Create the file `/var/qps/env_config/modules/custom/manifests/soap_http.pp` with the following contents on Cluster Manager.

Change ethX with the Unified API IP interface like eth0/eth1/eth2.

```
class custom::soap_http(
  $haproxytype = "-soaphttp",
)
{
  service { "haproxy-soaphttp":
    enable => false,
    require => [Package [ "haproxy" ],File ["/etc/haproxy/haproxy-soaphttp.cfg"],
File['/etc/init.d/haproxy-soaphttp'], Exec["sysctl_refresh"]],
  }
  file { "/etc/init.d/haproxy-soaphttp":
    owner => "root",
    group => "root",
    content => template('qps/etc/init.d/haproxy'),
    require => Package [ "haproxy" ],
    notify => Service['haproxy-soaphttp'],
```

```
    mode => 0744
  }
  file { "/etc/haproxy/haproxy-soaphttp.cfg":
    owner => "root",
    group => "root",
    content => template('custom/etc/haproxy/haproxy-soaphttp.erb'),
    require => Package [ "haproxy" ],
    notify => Service['haproxy-soaphttp'],
  }
  file { "/etc/monit.d/haproxy-soaphttp":
    content => template("custom/etc/monit.d/haproxy-soaphttp"),
    notify => Service["monit"],
  }
  exec { "remove ckconfig for haproxy-soaphttp":
    command => "/sbin/chkconfig --del haproxy-soaphttp",
    require => [Service['haproxy-soaphttp']],
  }
  firewall { '100 allow soap http':
    port => 8080,
    iniface => "ethX",
    proto => tcp,
    action => accept,
  }
}
```

Step 6 Validate the syntax of your newly created Puppet script on Cluster Manager:

```
/usr/bin/puppet parser validate /var/qps/env_config/modules/custom/manifests/soap_http.pp
```

Step 7 Rebuild your Environment Configuration on Cluster Manager:

```
/var/qps/install/current/scripts/build/build_env_config.sh
```

Step 8 Reinitialize your lb01/02 environments on Cluster Manager:

The following commands will take few minutes to complete.

```
ssh lb01 /etc/init.d/vm-init
ssh lb02 /etc/init.d/vm-init
```

Step 9 Validate SOAP request on http:

a) Verify the haproxy services are running on lb01 and lb02 by executing the commands on Cluster Manager:

```
ssh lb01 monit summary | grep haproxy-soaphttp
Process 'haproxy-soaphttp'                  Running
ssh lb01 service haproxy-soaphttp status
haproxy (pid  11061) is running...
ssh lb02 monit summary | grep haproxy-soaphttp
Process 'haproxy-soaphttp'                  Running
ssh lb02 service haproxy-soaphttp status
haproxy (pid  13458) is running...
```

b) Verify the following URLs are accessible:

```
Unified API WSDL: http://<IP address>:8080/ua/wsdl/UnifiedApi.wsdl
Unified API XSD: http://<IP address>:8080/ua/wsdl/UnifiedApi.xsd
```

where, <*IP address*> is the IP address set in Step 2, on page 63.

TACACS+

This section covers the following topics:

- Overview, on page 66
- TACACS+ Service Requirements, on page 66
- Caching of TACACS+ Users, on page 67

Overview

Cisco Policy Suite (CPS) is built around a distributed system that runs on a large number of virtualized nodes. Previous versions of the CPS software allowed operators to add custom accounts to each of these virtual machines (VM), but management of these disparate systems introduced a large amount of administrative overhead.

CPS has been designed to leverage the Terminal Access Controller Access Control System Plus (TACACS+) to facilitate centralized management of users. Leveraging TACACS+, the system is able to provide system-wide authentication, authorization, and accounting (AAA) for the CPS system.

Further the system allows users to gain different entitlements based on user role. These can be centrally managed based on the attribute-value pairs (AVP) returned on TACACS+ authorization queries.

TACACS+ Service Requirements

To provide sufficient information for the Linux-based operating system running on the VM nodes, there are several attribute-value pairs (AVP) that must be associated with the user on the ACS server used by the deployment. User records on Unix-like systems need to have a valid “passwd” record for the system to operate correctly. Several of these fields can be inferred during the time of user authentication, but the remaining fields must be provided by the ACS server.

A standard “passwd” entry on a Unix-like system takes the following form:

```
<username>:<password>:<uid>:<gid>:<gecos>:<home>:<shell>
```

When authenticating the user via TACACS+, the software can assume values for the username, password, and gecos fields, but the others must be provided by the ACS server. To facilitate this need, the system depends on the ACS server provided these AVP when responding to a TACACS+ Authorization query for a given username:

- uid

  A unique integer value greater than or equal to 501 that serves as the numeric user identifier for the TACACS+ authenticated user on the VM nodes. It is outside the scope of the CPS software to ensure uniqueness of these values.

- gid

  The group identifier of the TACACS+ authenticated user on the VM nodes. This value should reflect the role assigned to a given user, based on the following values:

  - `gid=501 (qns-su)`

This group identifier should be used for users that are entitled to attain superuser (or 'root') access on the CPS VM nodes.

- `gid=504 (qns-admin)`

  This group identifier should be used for users that are entitled to perform administrative maintenance on the CPS VM nodes.

  Note For stopping/starting the Policy Server (QNS) process on node, the qns-admin user should use `monit`:

  For example,

  ```
  sudo monit stop qns-1
  sudo monit start qns-1
  ```

- `gid=505 (qns-ro)`

  This group identifier should be used for users that are entitled to read-only access to the CPS VM nodes.

- home

  The user's home directory on the CPS VM nodes. To enable simpler management of these systems, the users should be configured with a pre-deployed shared home directory based on the role they are assigned with the gid.

  - home=/home/qns-su should be used for users in the `qns-su` group (gid=501)
  - home=/home/qns-admin should be used for users in the `qnsadmin` group (gid=504)
  - home=/home/qns-ro should be used for users in the `qns-ro` group (gid=505)

- shell

  The system-level login shell of the user. This can be any of the installed shells on the CPS VM nodes, which can be determined by reviewing the contents of `/etc/shells` on one of the CPS VM nodes. Typically, this set of shells is available in a CPS deployment:

  - /bin/sh
  - /bin/bash
  - /sbin/nologin
  - /bin/dash
  - /usr/bin/sudosh

  The `/usr/bin/sudosh` shell can be used to audit user's activity on the system.

Caching of TACACS+ Users

The user environment of the Linux-based VMs needs to be able to lookup a user's `passwd` entry via different columns in that record at different times. The TACACS+ NSS module provided as part of the CPS solution

however is only able to query the Access Control Server (ACS) for this data using the `username`. For this reason the system relies upon the Name Service Cache Daemon (NSCD) to provide this facility locally after a user has been authorized to use a service of the ACS server.

More details on the operations of NSCD can be found by referring to online help for the software (nscd --help) or in its man page (nscd(8)). Within the CPS solution it provides a capability for the system to lookup a user's `passwd` entry via their `uid` as well as by their `username`.

To avoid cache coherence issues with the data provided by the ACS server the NSCD package has a mechanism for expiring cached information.

The default NSCD package configuration on the CPS VM nodes has the following characteristics:

- Valid responses from the ACS server are cached for 600 seconds (10 minutes)
- Invalid responses from the ACS server (user unknown) are cached for 20 seconds
- Cached valid responses are reloaded from the ACS server 5 times before the entry is completely removed from the running set -- approximately 3000 seconds (50 minutes)
- The cache are persisted locally so it survives restart of the NSCD process or the server

It is possible for an operator to explicitly expire the cache from the command line. To do so the administrator need to get the shell access to the target VM and execute the following command as a root user:

```
# nscd -i passwd
```

The above command will invalidate all entries in the passwd cache and force the VM to consult with the ACS server for future queries.

There may be some unexpected behaviors of the user environment for TACACS+ authenticated users connected to the system when their cache entries are removed from NSCD. This can be corrected by the user by logging out of the system and logging back into it or by issuing the following command, which forces the system to query the ACS server:

```
# id -a "$USER"
```

Reading Log Files

Only qns-ro and qns-admin users are allowed to view log files at specific paths according to their role and maintenance requirement. Access to logs are allowed only using the following paths:

- /var/log/
- /var/log/broadhop/scripts/
- /var/log/httpd
- /var/log/redis
- /var/log/broadhop

Commands such as `cat`, `less`, `more`, and `find` cannot be executed using sudo in CPS 10.0.0 or higher releases.

To read any file, execute the following script using sudo:

```
$ sudo /var/qps/bin/support/logReader.py -r h -n 2 -f /var/log/puppet.log
```

where,

- `-r`: Corresponds to tail (t) ,tailf (tf), and head (h) respectively
- `-n`: Determines number of lines to be read. It works with the -r option. This is an optional parameter.
- `-f`: Determines the complete file path to be read.

Note

- Non-root users cannot view the sudosh logs.
- Support to read gunzipped files is also available.

CRD APIs

You use Custom Reference Data (CRD) APIs to query, create, delete, and update CRD table data without the need to utilize the Control Center interface. The CRD APIs are available via a REST interface.

Limitations

These APIs allow maintenance of the actual data rows in the table. They do not allow the creation of new tables or the addition of new columns. Table creation and changes to the table structure must be completed via the Policy Builder application.

Table names must be all in lowercase alphanumeric to utilize these APIs. Neither spaces nor special characters are allowed in the table name.

- Table names containing uppercase characters will return code 400 Bad Request.
- Spaces in the name are also not allowed and will be flagged as an error in Policy Builder.
- Special characters even when escaped or encoded in ASCII cause problems with the APIs and should not be used.

Setup Requirements

Policy Server

The feature com.broadhop.custrefdata.service.feature needs to be installed on the Policy Server.

In a High Availability (HA)/Distributed CPS deployment, this feature should be installed on the QNS0x nodes.

Policy Builder

The feature `com.broadhop.client.feature.custrefdata` needs to be installed in Policy Builder.

Step 1 Login into Policy Builder.

Step 2 Select **Reference Data** tab.

Step 3 From the left pane, select **Systems**.

Step 4 Select and expand your system name.

Step 5 Select **Plugin Configurations** (or a sub cluster or instance), a Custom Reference Data Configuration plugin configuration is defined.

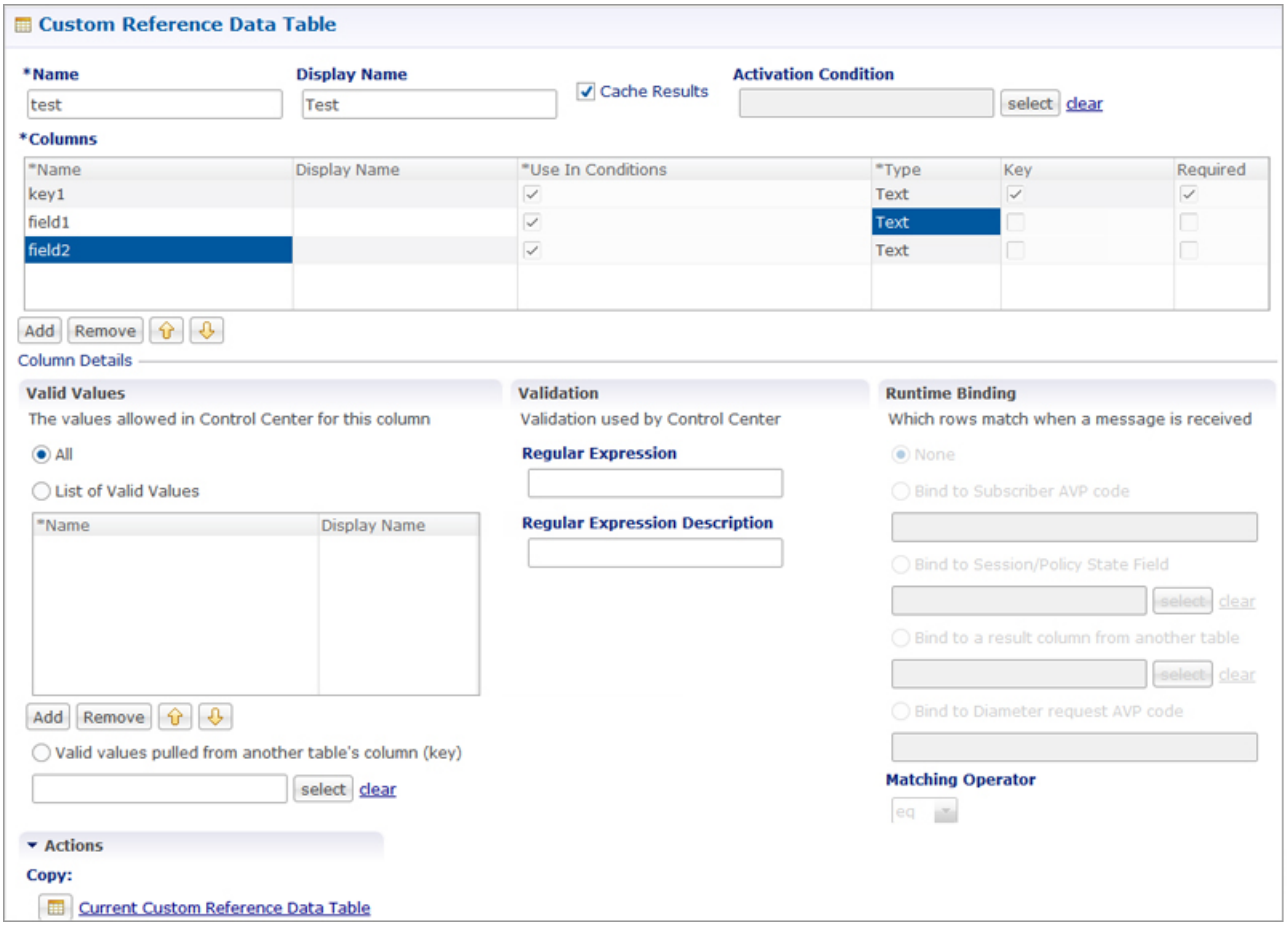
The following parameters can be configured under **Custom Reference Data Configuration**:

Table 8: Custom Reference Data Configuration

Parameter	Description
Primary Database IP Address	IP address of the primary sessionmgr database.
Secondary Database IP Address	Optional, this field is the IP address of a secondary, backup, or failover sessionmgr database.
Database Port	Port number of the sessionmgr. It should be the same for both the primary and secondary databases.
Db Read Preference	Read preference describes how sessionmgr clients route read operations to members of a replica set. You can select from the following drop-down list: • Primary: Default mode. All operations read from the current replica set primary. • PrimaryPreferred: In most situations, operations read from the primary but if it is unavailable, operations read from secondary members. • Secondary: All operations read from the secondary members of the replica set. • SecondaryPreferred: In most situations, operations read from secondary members but if no secondary members are available, operations read from the primary. For more information, refer to http://docs.mongodb.org/manual/core/read-preference/.
Connection Per Host	Number of connections that are allowed per database host. Default value is 100.

Step 6 In **Reference Data** tab > **Custom Reference Data Tables**, at least one Custom Reference Data Table must be defined.

Figure 10: Custom Reference Data Table



The following parameters can be configured under Custom Reference Data Table:

Table 9: Custom Reference Data Table Parameters

Parameter	Description
Name	This is the name of the table that will be stored in the database. It should start with alphanumeric characters, should be lowercase OR uppercase but not MixedCase, and should not start with numbers, no special characters are allowed, use "_" to separate words. For example, logical_apn = GOOD, logicalAPN = BAD, no_spaces. For more information, refer to Limitations, on page 69.
Display Name	This is the name of the table that will be displayed in Control Center.
Cache Results	This indicates whether the tables should be cached in memory. This should be checked for production. For more information, refer to Caching, on page 74.

Parameter	Description
Activation Condition	This is the Custom Reference Data Trigger which needs to be true before evaluating this table. This can be used to have multiple tables create the same data depending on conditions or to improve performance if tables don't need to be evaluated based on an initial condition(s).
Best Match	If checked, this allows '*' to be used in the values of the data and the best matching row is returned.
Evaluation Order	This indicates the order the tables within the search table group should be evaluated. Starting with 0 and increasing.

Columns	Columns correspond to the 'schema' for each column we're creating for this Custom Reference Data table.
	• Name: The name of the column in the database.
	• Display Name: A more readable display name.
	• Use In Conditions: This represents whether this row will be available for conditions in Policies or Use Case Templates. There is a performance cost to having these checked, so we recommend to uncheck unless they are required.
	Default value is checked (true).
	• Type: the type determines what values will be allowed when creating them in control center.
	  • Text: The value is allowed to be any characters. For example, example123!.
	  • Number: The value is allowed to be any whole number. For example, 1234.
	  • Decimal: The value is allowed to be any number (including decimals). For example, 1.234.
	  • True/False: The value needs to be 'true' or 'false'. For example, true.
	  • Date: The value is a date without a time component (May 17th, 2020).
	  • DateTime: The value is a date + time (May 17th, 2020 5:00pm).
	• Key: This indicates that this column is all or part of the 'key' for the table that makes this row unique. By default, a key is required. Keys also are allowed set the Runtime Binding fields to populate this data from the current message/session. Typically, keys are bound to data from the current session (APN, RAT Type) and other values are derived from them. Keys can also be set to a value derived from another Custom Reference Data table.
	• Required: This indicates whether this field will be marked required in Control Center. A key is always required.

Parameter	Description
Valid Values	These are the valid values which will be allowed in Control Center (creates a list box). • List of Valid Values: A list of name/display name pairs which will be used to create the list. Valid values can also contain a 'name' which will be the actual value of the column and a display value which allows Control Center to display an easier to use name. • Valid Values pulled from another Table: This allows initializing the list based on another Custom Reference Data table. The 'name' value will be pulled from another table. There is no way to customize a 'display' name in this manner.
Validation	The validation set here will be checked by Control Center before allowing a row to be added. • Regular Expression: This is the Java regular expression that will be run on the proposed new cell value to validate it as described in http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html. • Regular Expression Description: This is a message to the user indicating what the regular expression is trying to check.

Runtime Binding	Runtime binding is how key column data gets filled out ('bound') from data in the current session. There are multiple ways to bind this data and it is also possible to set an operator to define what should match (equals, less than, etc).
	• Bind to Subscriber AVP Code: This pulls the value from an AVP on the subscriber. It will also pull values from a session AVP or a Policy Derived AVP.
	• Bind to Session/Policy State Field: This pulls the value from a Policy State Data Retriever which knows how to retrieve a single value for a session
	• Bind to a Result Column from another Table: This allows the key to be filled out from a columns value from another table. This allows 'normalizing' the table structure and not having on giant table with a lot of duplicated values.
	• Bind to Diameter Request AVP code: This allows the key be filled out from an AVP on the Diameter request.
	• Matching Operator: This allows the row to be 'matched' in other ways than having the value be 'equals'. Default value is equals.
	  • eq: Equal
	  • ne: Not Equal
	  • gt: Greater than
	  • gte: Greater than or equal
	  • lt: Less than
	  • lte: Less than or equal

Architecture

MongoDB

The MongoDB database containing the CRD tables and the data is located in the MongoDB instance specified in the CRD plugin configuration.

The database is named cust_ref_data.

Two system collections exist in that database and do not actually contain CRD data:

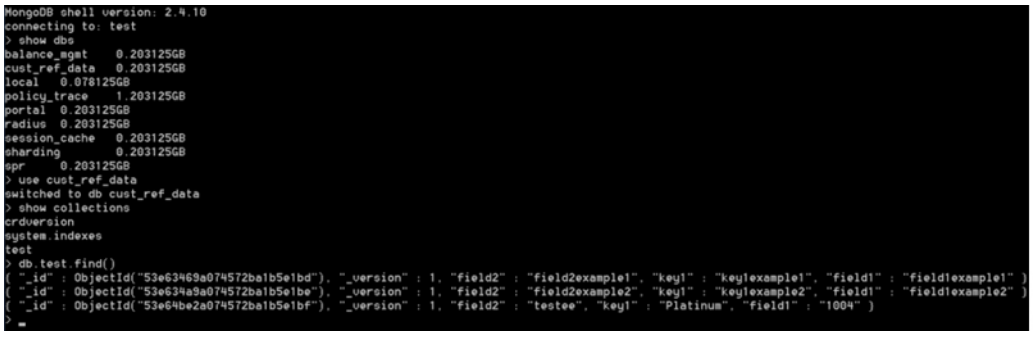
- system.indexes — used by MongoDB. These are indices set on the database.
- crdversion — contains a document indicating the version of all the CRD tables you have defined. The version field increments by 1 every time you make a change or add data to any of your CRD tables.

A collection is created for each CRD table defined in Policy Builder.

- This collection contains a document for each row you define in the CRD table.
- Each document contains a field for each column you define in the CRD table.
- The field contains the value specified for the column for that row in the table.
- Additionally, there is a _id field which contains the internal key used by MongoDB and _version which is used by CPS to provide optimistic locking protection, essentially to avoid two threads overwriting the other's update, on the document.

An example is shown below:

Figure 11: CRD Table in Policy Builder



Caching

Setting the Cache Results to true (checked) is the default and recommended settings in most cases as it yields the best performance. Use of the cached copy also removes the dependency on the availability of the CRD database, so if there is an outage or performance issue, policy decisions utilizing the CRD data won't be impacted.

The cached copy of the table is refreshed on CPS restart and whenever the API writes a change to the CRD table, otherwise the cached copy is used and the database is not accessed.

API Endpoints and Examples

The URL used to access the CRD API are different depending on the type of deployment (High Availability or All-in-One):

High Availability (HA): https://<lbvip01>:8443/custrefdata/<tablename>/_<operation>

All-In-One (AIO): http://<ip>:8080/custrefdata/<tablename>/_<operation>

The examples in the following sections refer to the HA URL.

Query API

Purpose

Returns all rows currently defined in the specified table.

HTTP Operation Type

GET

Example URL

https://<lbvip01>:8443/custrefdata/test/_query

Example URL with Filtering

https://<lbvip01>:8443/custrefdata/test/_query?key1=Platinum

Payload

None, although parameters can be specified on the URL for filtering.

Response

Success returns code 200 Ok; XML indicating rows defined is returned. If there are no records in the table, 200 Ok is returned with empty rows in it.

If the table does not exist, code 400 Bad Request is returned.

Example Response without Filtering

```
<rows>
  <row>
    <field code=”field1” value=”1004”/>
    <field code=”field2” value=”testee”/>
    <field code=”key1” value=”Platinum”/>
  </row>
  <row>
    <field code=”field1” value=”1004”/>
    <field code=”field2” value=”testee”/>
    <field code=”key1” value=”Platinum99”/>
  </row>
  <row>
    <field code=”field1” value=”field1example1”/>
    <field code=”field2” value=”field2example1”/>
    <field code=”key1” value=”key1example1”/>
  </row>
```

```
  <row>
    <field code="field1" value="field1example2"/>
    <field code="field2" value="field2example2"/>
    <field code="key1" value="key1example2"/>
  </row>
</rows>
```

Example Response with Filtering

```
<rows>
<rows>
  <row>
    <field code="field1" value="1004"/>
    <field code="field2" value="testee"/>
    <field code="key1" value="Platinum"/>
  </row>
</rows>
```

The response returns keys with the tag "field code". If you want to use the output of Query as input to one of the other APIs, the tag needs to be changed to "key code". Currently using "field code" for a key returns code 404 Bad Request and a java.lang.NullPointerException.

Create API

Purpose

Create a new row in the specified table.

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/test/_create

Example Payload

```
<row>
  <key code="key1" value="Platinum"/>
  <field code="field1" value="1004"/>
  <field code="field2" value="testee"/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. The key cannot already exist for another row; submission of a duplicate key returns code 400 Bad Request.

If creating a row fails, API returns 400 Bad Request.

Note Create API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Create operation is not allowed for subversion table

Update API

Purpose

Updates the row indicated by the key code in the table with the values specified for the field codes.

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/test/_update

Example Payload

```
<row>
  <key code="key1" value="Platinum"/>
  <field code="field1" value="1005"/>
  <field code="field2" value="tester"/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. The key cannot be changed. Any attempt to change the key returns code 404 Not Found.

If updating a row fails, API returns 400 Bad Request.

Note Update API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Update operation is not allowed for subversion table

Delete API

Purpose

Removes the row indicated by the key code from the table.

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/test/_delete

Example Payload

```
<row>
<key code="key1" value="Platinum"/>"/>
</row>
```

Response

Success returns code 200 Ok; no data is returned. If the row to delete does not exist, code 404 Not Found is returned.

If deleting a row fails, API returns 400 Bad Request.

Note Delete API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Delete operation is not allowed for subversion table

Data Comparison API

Purpose

Determines whether the same CRD table data content is being used at different data centers.

The following three optional parameters can be provided to the API:

- tableName: Returns the checksum of a specified CRD table `tableName` indicating if there is any change in the specified table. If the value returned is same on different servers, it means there is no change in the configuration and content of that table.
- includeCrdversion: Total database checksum contains combination of checksum of all CRD tables configured in Policy Builder. If this parameter is passed as true in API, then total database checksum includes the checksum of "crdversion" table. Default value is false.
- orderSensitive: Calculates checksum of the table by utilizing the order of the CRD table content. By default, it does not sort the row checksums of the table and returns order sensitive checksum of every CRD table. Default value is true.

custrefdata/_checksum

Database level Checksum API returns checksum details for all the CRD tables and the database. If the value returned is same on different servers, there will be no change in the configuration and content of any CRD table configured in Policy Builder.

HTTP Operation Type

GET

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_checksum

Response

```
<response>
    <checksum><all-tables-checksum></checksum>
    <tables>
        <table name="<table-1-name>" checksum="<checksum-of-table-1>"/>
        <table name="<table-2-name>" checksum="<checksum-of-table-2>"/>
```

```
        <table name="<table-n-name>" checksum="<checksum-of-table-n>"/>
    </tables>
</response>
```

/custrefdata/_checksum?tableName=<user-provided-table-name>

Table specific Checksum API returns the checksum details for the specific CRD table. If the value returned is same on different servers, there will be no change in the configuration and content of that table.

HTTP Operation Type

GET

Example Endpoint URL

https://<lbvip01>:8443 /custrefdata/_checksum?tableName=<user-provided-table-name>

Response

```
<response>
    <tables>
        <table name="<user-provided-table-name>" checksum="<checksum-of-specified-table"/>
    </tables>
</response>
```

Note Table specific Checksum API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Checksum operation is not allowed for subversion table

Table Drop API

Purpose

Drops custom reference table from MongoDB to avoid multiple stale tables in the system.

The Table Drop API is used in the following scenarios:

- If a CRD table does not exist in Policy Builder but exists in the database, the API can be used to delete the table from the database.
- If a CRD table exists in Policy Builder and database, the API cannot delete the table from the database. If this is attempted the API will return an error: “Not permitted to drop this table as it exists in Policy Builder”.
- If a CRD table does not exist in Policy Builder and database, the API will also return an error `No table found:<tablename>`.

/custrefdata/<table_name>/_drop

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/<table_name>/_drop

Note Drop API does not support SVN CRD table operations and displays the following error message when Snv Crd Data checkbox is enabled in CRD table configuration:

Drop operation is not allowed for subversion table

Export API

Purpose

Exports single and multiple CRD table and its data.

/custrefdata/_export?tableName=<table_name>

Exports single CRD table and its data.

Returns an archived file containing csv file with information of specified CRD table `table_name`.

HTTP Operation Type

GET

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_export?tableName=<table_name>

/custrefdata/_export

Exports all CRD tables and its data.

Returns an archived file containing csv file with information for each CRD Table.

HTTP Operation Type

GET

Example Endpoint URL

https://<lbvip01>:8443 /custrefdata/_export

Note Export API does not support Svn CRD tables and displays the following warning message in the Response Header "Export-Warning":

Datasource for tables [table1, table2,...] is subversion. Response will not contain data for these tables and skipped SVN CRD tables to be a part of archive.

Import API

Purpose

Imports CRD table and its data.

It takes an archived file as an input which contains one or more csv files containing CRD tables information.

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_import

https://<lbvip01>:8443/custrefdata/_import?batchOperation=true

https://<lbvip01>:8443/custrefdata/_import?batchOperation=false&duplicateValidation=true

Note

1. The "batchOperation" flag is used to insert CRD data in the batch. The default value is true and if you do not provide it in the request parameter the default value is taken.
2. The "duplicateValidation" flag is used to validate or invalidate duplicate data in the archive. The default value is true and if you do not provide it in the request parameter the default value is taken which means it will always validate your data as duplicate.
3. If "batchOperation" is true, the API will validate your data as duplicate data regardless of the value provided for "duplicateValidation".

Note Import API supports SVN CRD table operations in the following scenarios:

- If the archive contains only mongodb tables, success message is displayed in the response.
- If the archive contains only SVN tables, success and warning messages are displayed in the response.
- If the archive contains both mongodb and SVN tables, success and warning messages are displayed in the response.

Import Single File API

Purpose

Imports bulk CRD data by sending any supported file in the API request.

Supports only CSV and XLS file formats.

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_importsinglefile

Note

1. Error responses are thrown in the following scenarios:
   - When the attached file is of a different format other than CSV and XLS.
   - When an empty file is attached.
   - When the attached file has wrong headers.
   - When the attached file does not have the same file as that of the Policy Builder table name.
   - When the attached file has duplicate records.
   - When no file is attached.
2. Ensure your .xls file does not contain any extra empty and colored header. If the .xls file contains any colored and empty header (header with color but no title), it is considered as a part of the Policy Builder table column. During import file operation, this type of header causes the API to send `Mismatch found between imported csv headers and policy builder table columns` error in response. This is because the empty header is considered as a column from Policy Builder but the Policy Builder table does not contain this empty column.
3. Import Single File API does not support import of SVN CRD table data and displays the following error message:

   Single file import is not allowed for subversion table

Snapshot POST API

Purpose

Creates a snapshot of the CRD tables on the system. The created snapshot will contain CRD table data, policy configuration and checksum information for all CRD tables.

/custrefdata/_snapshot?userId=<user_id>&userComments=<user_comments>

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_snapshot?userId=<user_id>&userComments=<user_comments>

Optional Parameters

userComments

Note Snapshot POST API does not support export of the contents of Svn CRD tables. The API returns the following warning message if there are any Svn CRD tables present while creating snapshot:

Datasource for tables [table_1, table_2…] is subversion. Data for these tables will not come from database (mongodb)

Snapshot GET API

Purpose

Enables you to get the list of all valid snapshots in the system.

The following information is available in the list of snapshots:

- Snapshot name
- Snapshot path
- Date and time of snapshot creation
- User comments provided on creation of the snapshot
- Checksum information of CRD tables
- Policy configuration SVN version number

/custrefdata/_snapshot

HTTP Operation Type

GET

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_snapshot

Example Response

```
<snapshots>
    <snapshot>
        <name><date-and-time>_<user-id></name>
        <snapshotPath>/var/broadhop/snapshot/20160620011825306_qns</snapshotPath>
        <creationDateAndTime>20/06/2016 01:18:25:306</creationDateAndTime>
        <comments>snapshot-1 june</comments>
        <policyVersion>903</policyVersion>
        <checksum checksum="60f51dfd4cd4554910da44a776c66db1">
            <table name=<table-name-1> checksum="<table-checksum-1>"/>
                …
            <table name=<table-name-n> checksum="<table-checksum-n>"/>
        </checksum>
    </snapshot>
    <snapshot>
                …
    </snapshot>
</snapshots>
```

Note Snapshot GET API does not return checksum information of Svn CRD tables as they are not part of created snapshots.

Revert API

Purpose

Enables you to revert the CRD data to a specific snapshot. If the specific snapshot name is not provided, the API will revert to the latest snapshot.

/custrefdata/_revert?snapshotName=<snapshot_name>

HTTP Operation Type

POST

Example Endpoint URL

https://<lbvip01>:8443/custrefdata/_revert?snapshotName=<snapshot_name>

Optional Parameter

snapshotName

Note Revert API does not support reverting of CRD data for Svn CRD tables. For Svn CRD table, it clears the mongodb table and displays the following warning message:

Datasource for tables [table_1, table_2…] is subversion. Data for these tables will be reverted using svn datasource not from database (mongodb)

Tips for Usage

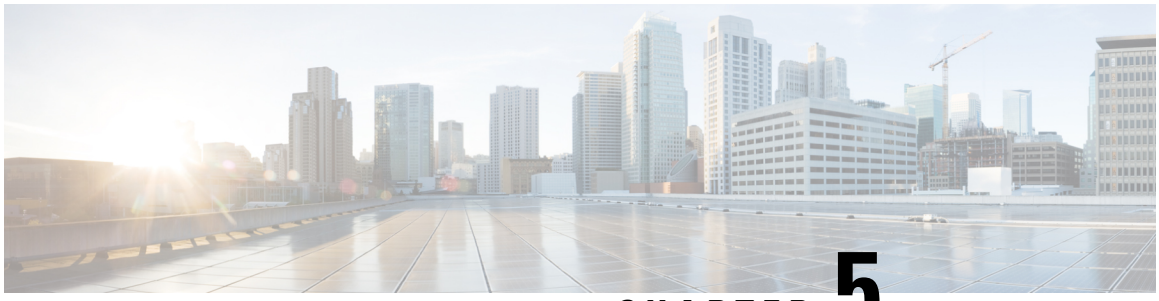
The Query API is a GET operation which is the default operation that occurs when entering a URL into a typical web browser.

The POST operations, Create, Update, and Delete, require the use of a REST client so that the payload and content type can be specified in addition to the URL. REST clients are available for most web browsers as plug-ins or as part of web service tools, such as SoapUI. The content type when using these clients should be specified as application/xml or the equivalent in the chosen tool.

View Logs

You can view the API logs in the OAM (pcrfclient) VM at the following location:

```
/var/log/broadhop/consolidated-qns.log
```



CHAPTER 5

Tracking CPS GUI and API Usage

- Track Usage, on page 87

Track Usage

Use the Audit History to track usage of the various GUIs and APIs.

If enabled, each request is submitted to the Audit History database for historical and security purposes. The user who made the request, the entire contents of the request and if it is subscriber-related (a network ID value), all network IDs are also stored in a searchable field.

Capped Collection

By default, the Audit History uses a 1 GB capped collection in MongoDB. The capped collection automatically removes documents when the size restriction threshold is hit. The oldest document is removed as each new document is added. For customers who want more than 1 GB of audit data, contact the assigned Cisco Advanced Services Engineer to get more information.

Configuration in Policy Builder is done in GB increments. It is possible to enter decimals, for example, 9.5 will set the capped collection to 9.5 GB.

PurgeAuditHistoryRequests

When using a capped collection, MongoDB places a restriction on the database and does not allow the deletion of data from the collection. Therefore, the entire collection must be dropped and re-created. This means that the PurgeAuditHistory queries have no impact on capped collections.

AuditRequests

As a consequence of the XSS defense changes to the API standard operation, any XML data sent in an AuditRequest must be properly escaped even if inside CDATA tags.

For example, <ExampleRequest>...</ExampleRequest>

For more information on AuditType, refer to Cisco Policy Suite Unified API 2.3.0 Guide.

Operation

By default, Audit History is ON but it can be turned OFF.

- `ua.client.submit.audit=true` — property used by Policy Builder and set in /etc/broadhop/pb/pb.conf
- `Submit Requests to Audit Log` — Unified API plug-in configuration in Policy Builder.

Initial Setup

There are three parts to the Audit History:

- Server — database and Unified API
- Policy Builder
- Audit Client — bundle that the Policy Builder uses to send Audit requests

Step 1 Start the Policy Builder with the following property:

`-Dua.client.submit.audit=false` (set in `/etc/broadhop/pb/pb.conf`)

Step 2 Add and configure the appropriate plug-in configurations for Audit History and Unified API.

Step 3 Publish the Policy Builder configuration.

Step 4 Start the CPS servers.

Step 5 Restart the Policy Builder with the following property:

```
-Dua.client.submit.audit=true
```

```
-Dua.client.server.url=https://lbvip02:8443/ua/soap
```

or

```
-Dua.client.server.url=http://lbvip02:8080/ua/soap
```

Read Requests

The Audit History does not log read requests by default.

- GetRefDataBalance
- GetRefDataServices
- GetSubscriber
- GetSubscriberCount
- QueryAuditHistory
- QueryBalance
- QuerySession

- QueryVoucher
- SearchSubscribers

The Unified API also has a Policy Builder configuration option to log read requests which is set to false by default.

APIs

All APIs are automatically logged into the Audit Logging History database, except for QueryAuditHistory and KeepAlive. All Unified API requests have an added Audit element that should be populated to provide proper audit history.

Querying

The query is very flexible - it uses regex automatically for the id and dataid, and only one of the following are required: id, dataid, or request. The dataid element typically will be the networkId (Credential) value of a subscriber.

Note Disable Regex. The use of regular expressions for queries can be turned off in the Policy Builder configuration.

The id element is the person or application who made the API request. For example, if a CSR log into Control Center and queries a subscriber balance, the id will be that CSR's username.

The dataid element is typically the subscriber's username. For example, if a CSR log into Control Center and queries a subscriber, the id will be that of CSR's username, and the dataid will be the subscriber's credential (networkId value). For queries, the dataid value is checked for spaces and then tokenized and each word is used as a search parameter. For example, “networkId1 networkId2” is interpreted as two values to check.

The fromDate represents the date in the past from which to start the purge or query. If the date is null, the api starts at the oldest entry in the history.

The toDate represents the date in the past to which the purge or query of data includes. If the date is null, the api includes the most recent entry in the purge or query.

Purging

By default, the Audit History database is capped at 1 GB. Mongo provides a mechanism to do this and then the oldest data is purged as new data is added to the repository. There is also a PurgeAuditHistory request which can purge data from the repository. It uses the same search parameters as the QueryAuditHistory and therefore is very flexible in how much or how little data is matched for the purge.

Note Regex Queries! Be very careful when purging records from the Audit History database. If a value is given for dataid, the server uses regex to match on the dataid value and therefore will match many more records than expected. Use the QueryAuditHistory API to test the query.

Purge History

Each purge request is logged after the purge operation completes. This ensures that if the entire repo is destroyed, the purge action that destroyed the repo will be logged.

Control Center

The Control Center version 2.0 automatically logs all requests.

PurgeAuditHistoryRequest

This API purges the Audit History.

The query is very flexible - it uses regex automatically for the id and dataid, and only one of the following are required: id, dataid, or request. The dataid element typically will be the networkId (Credential) value of a subscriber.

The id element is the person or application who made the API request. For example, if a CSR logs into Control Center and queries a subscriber balance, the id will be that CSR's username.

The dataid element is typically the subscriber's username. For example, if a CSR logs into Control Center and queries a subscriber, the id will be that CSR's username, and the dataid will be the subscriber's credential (networkId value). For queries, the dataid value is checked for spaces and then tokenized and each word is used as a search parameter. For example, “networkId1 networkId2” is interpreted as two values to check.

The fromDate represents the date in the past from which to start the purge or query. If the date is null, the api starts at the oldest entry in the history.

The toDate represents the date in the past to which the purge or query of data includes. If the date is null, the api includes the most recent entry in the purge or query.

Note Size-Capped Database

If the database is capped by size, then the purge request ignores the request key values and drops the entire database due to restrictions of the database software.

Schema

```
<PurgeAuditHistoryRequest>
<key> AuditKeyType </key> [1]
</PurgeAuditHistoryRequest>
```

Example

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <PurgeAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>username</id>
        <dataid>subscriber</dataid>
        <request>API Name</request>
        <fromDate>2011-01-01T00:00:00Z</fromDate>
        <toDate>2011-01-01T00:00:00Z</toDate>
      </key>
    </PurgeAuditHistoryRequest>
```

```
  </se:Body>
</se:Envelope>
```

To purge all CreateSubscriberRequest:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <PurgeAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <request>CreateSubscriberRequest</request>
      </key>
    </PurgeAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

To purge all CreateSubscriberRequest by CSR:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <PurgeAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>csrusername</id>
        <request>CreateSubscriberRequest</request>
      </key>
    </PurgeAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

To purge all actions by CSR for a given subscriber for a date range:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <PurgeAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>csrusername</id>
        <dataid>subscriber@gmail.com</dataid>
        <fromDate>2010-01-01T00:00:00Z</fromDate>
        <toDate>2012-11-01T00:00:00Z</toDate>
      </key>
    </PurgeAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

QueryAuditHistoryRequest

This API queries the Audit History.

The query is very flexible - it uses regex automatically for the id and dataid, and only one of the following are required: id, dataid, or request. The dataid element typically will be the networkId (Credential) value of a subscriber.

The id element is the person or application who made the API request. For example, if a CSR logs into Control Center and queries a subscriber balance, the id will be that CSR's username.

The dataid element is typically the subscriber's username. For example, if a CSR logs into Control Center and queries a subscriber, the id will be that CSR's username, and the dataid will be the subscriber's credential (networkId value). For queries, the dataid value is checked for spaces and then tokenized and each word is used as a search parameter. For example, "networkId1 networkId2" is interpreted as two values to check.

The fromDate represents the date in the past from which to start the purge or query. If the date is null, the api starts at the oldest entry in the history.

The toDate represents the date in the past to which the purge or query of data includes. If the date is null, the api includes the most recent entry in the purge or query.

Schema:

```
<QueryAuditHistoryRequest>
<key> AuditKeyType </key> [1]
</QueryAuditHistoryRequest>
```

Example:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <QueryAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>username</id>
        <dataid>subscriber</dataid>
        <request>API Name</request>
        <fromDate>2011-01-01T00:00:00Z</fromDate>
        <toDate>2011-01-01T00:00:00Z</toDate>
      </key>
    </QueryAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

To find all CreateSubscriberRequest:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <QueryAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <request>CreateSubscriberRequest</request>
      </key>
    </QueryAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

To find all CreateSubscriberRequest by CSR:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <QueryAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>csrusername</id>
        <request>CreateSubscriberRequest</request>
      </key>
    </QueryAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

To find all actions by CSR for a given subscriber for a date range:

```
<se:Envelope xmlns:se="http://schemas.xmlsoap.org/soap/envelope/">
  <se:Body>
    <QueryAuditHistoryRequest xmlns="http://broadhop.com/unifiedapi/soap/types">
      <key>
        <id>csrusername</id>
        <dataid>subscriber@gmail.com</dataid>
        <fromDate>2010-01-01T00:00:00Z</fromDate>
```

```
        <toDate>2012-11-01T00:00:00Z</toDate>
      </key>
    </QueryAuditHistoryRequest>
  </se:Body>
</se:Envelope>
```

Policy Builder

The Policy Builder automatically logs all save operations (Publish and Save to Client) to the Audit History database and also to a log file.

- Policy Builder Publish submits an entry to the Audit Logging Server (goes to database).
- Policy Builder Save to Client Repository submits an entry to the Audit Logging Server (goes to database).
- Whenever a screen is saved locally (Save button) XML is generated and logged for that user in `/var/log/broadhop/qns-pb.log`.

Example log in `qns-pb.log` from Local Save in Policy Builder:

```
2013-02-06 11:57:01,214 [UIThread [vt75cjqhk7v4noguyc9c7shp]] DEBUG
c.b.c.r.BroadhopResourceSetAudit -
Audit: Local file change made by: broadhop. Updated File:
file:/var/broadhop/pb/workspace/tmp-ITC2/checkout/ConfiguredExtensionPoint-43730cd7-b238-4b29-a828-d9b4
47e5a64f-33851.xmi
```

XML Representation of changed screen:

```
<?xml version="1.0" encoding="UTF-8"?>
<policy:ConfiguredExtensionPoint xmlns:policy="http://broadhop.com/policy"
id="43730cd7-b238-4b29-a828-d9b447e5a64f-33851">
  <extensionPoint
    href="virtual:URI#_vxG4swK1Ed-M48DL9vicxQ"/>
  <policies
    href="Policy-default-_sY__4L_REeGCdakzuzzlAg.xmi#_sY__4L_REeGCdakzuzzlAg"/>
</policy:ConfiguredExtensionPoint>
```

Controlling Local Save output:

In the logback.xml file that controls Policy Builder logging, add `com.broadhop.client.resourceset.BroadhopResourceSetAudit` as a category and set it to the desired level.

Reporting

For reporting purposes the following is the database structure in Mongo:

```
{
  "_id" :
  ObjectId("5097d75be4b0d5f7ab0d90fe"),
  "_id_key" :
  "username",
  "comment_key" :
  "comment",
"data_id_key" : [
  "networkId11921" ],
"timestamp_key" :
  ISODate("2012-11-05T15:12:27.673Z"),
  "request_key" :
  "DeleteQuotaRequest",
```

```
  "data_key" :
"<DeleteQuotaRequest><audit><id>username</id></audit><networkId><![CDATA
[networkId11921]]></networkId><balanceCode>DATA</balanceCode><code>Recurring</code>
<hardDelete>false</hardDelete></DeleteQuotaRequest>
"}
```

The following table describes the various Reporting Keys.

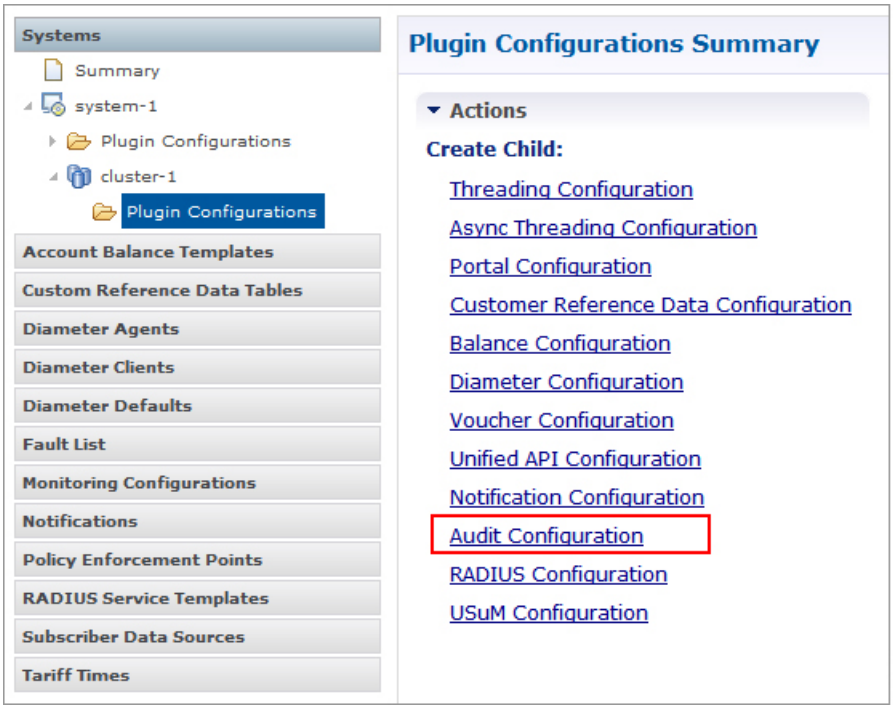
Table 10: Reporting Keys

Field	Description
_id	The database unique identifier.
_id_key	the username of person who performed the action. In the above example the CSR who issued the debit request.
comment_key	Some description of the audit action.
data_id_key	The credential of the subscriber. It is a list and so, if the subscriber has multiple credentials, then they will all appear in this list. Please note that, it is derived from the request data and so, for a CreateSubscriber request, there may be multiple credentials sent in the request and each will be saved in the data_id_key list. In the DebitRequest case, only one credential is listed because the request only has the single networkId field.
timestamp_key	The time the request was logged. If the timestamp value is null in the request then the Audit module automatically populates this value.
request_key	The name of the request. This provides a way to search on type of API request.
data_key	The actual request XML.

Audit Configuration

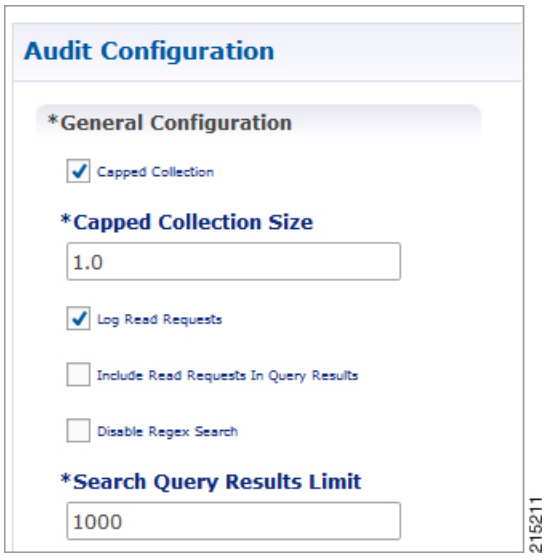
Step 1 Click the **Reference Data** tab, and then click **Systems** > *system name* > **Plugin Configurations**.

Figure 12: Plugin Configurations Summary



Step 2 Click **Audit Configuration** in the right pane to open the **Audit Configuration** dialog box.

Figure 13: Audit Configuration dialog box



Step 3 Under **Audit Configuration** there are different panes: **General Configuration**, **Queue Submission Configuration**, **Database Configuration**, and **Shard Configuration**. An example configuration is provided in the following figures:

Figure 14: Queue Submission Configuration pane

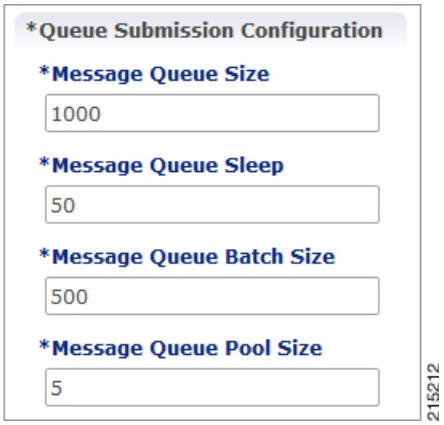


Figure 15: Database Configuration pane

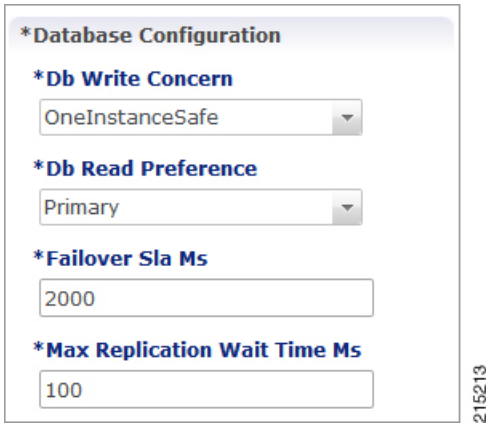
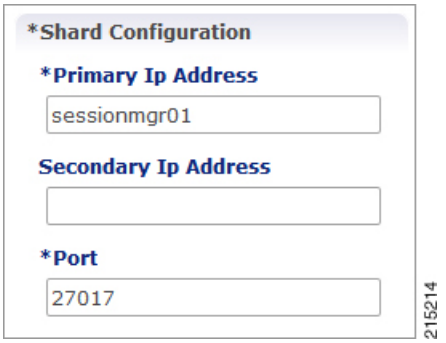


Figure 16: Shard Configuration pane



The following parameters are used to size and manage the internal queue that aids in the processing of Audit messages. The application offloads message processing to a queue to speed up the response time from the API.

Table 11: Audit Configuration Parameters

Parameter	Description
General Configuration	
Capped Collection	Select this check-box to activate capped collection function.
Capped Collection Size	By default, the Audit History uses a 1 GB capped collection in MongoDB. The capped collection automatically removes documents when the size restriction threshold is hit. Configuration in Policy Builder is done in GB increments. It is possible to enter decimals, for example, 9.5 will set the capped collection to 9.5 GB.
Log Read Requests	Select this check-box if you want read requests to be logged.
Include Read Requests in Query Results	Select this check-box only if you want to include read requests to be displayed in query results.
Disable Regex Search	If you select this check-box, the use of regular expressions for queries is turned off in the Policy Builder configuration.
Search Query Results Limit	This parameter limits the search results.
Queue Submission Configuration	
Message Queue Size	Total number of messages the queue can hold at any given time.
Message Queue Sleep	The amount of time for the runnable to sleep between batch processing. The time is in milliseconds.
Message Queue Batch Size	The number of messages to process in a given wake cycle.
Message Queue Pool Size	The number of threads in the execution pool to handle message processing.
Database Configuration	
Db Write Concern	Controls the write behavior of sessionMgr and for what errors exceptions are raised. Default option is OneInstanceSafe.
Db Read Preference	Read preference describes how sessionMgr clients route read operations to members of a replica set. The recommended option is typically Secondary Preferred. http://docs.mongodb.org/manual/core/read-preference/
Failover Sla Ms	This parameter is used to enter the amount of time to wait before starting failover database handling. The time is in milliseconds.

Parameter	Description
Max Replication Wait time Ms	This option specifies a time limit, in milliseconds, for the write concern. This parameter is applicable only if you select TwoInstanceSafe in Db Write Concern. This parameter causes write operations to return with an error after the specified limit, even if the required write concern eventually succeeds. When these write operations return, MongoDB does not undo successful data modifications performed before the write concern exceeded the replication wait time limit. This time is in milliseconds.
Shard Configuration	
Primary Ip Address	The IP address of the sessionmgr node hosting the Audit database.
Secondary Ip Address	The IP address of the sessionmgr node that provides fail over support for the primary database. This is the mirror of the database specified in the Primary IP Address field. Use this only for replication or replica pairs architecture. This field is present but deprecated to maintain backward compatibility.
Port	Enter the Port number of the Audit database as defined in /etc/broadhop/mongoConfig.cfg. The default value in Policy Builder is 27017. For All-In-One deployments, the default Audit database port number is configured as 27017 (no update is needed to this field). For HA or GR deployments, the default Audit database port is 27725. You must update this field to match the Audit database port (27725) or as defined in /etc/broadhop/mongoConfig.cfg.

According to your network requirements, configure the parameters in Audit Configuration and save the configuration.

Pre-configured auditd

In the `/usr/share/doc/audit-version/` directory, the audit package provides a set of pre-configured rules files.

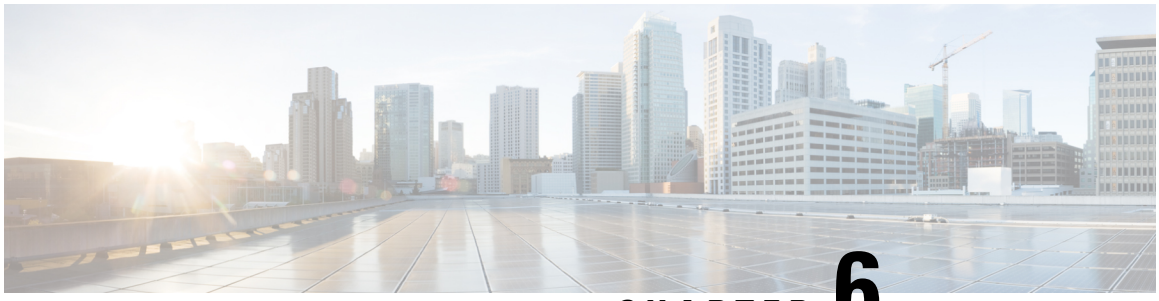
The Linux Audit system provides a way to track security-relevant information on your system. Based on pre-configured rules, Audit generates log entries to record as much information about the events that are happening on your system as possible.

In the `/usr/share/doc/audit-version/` directory, the audit package provides a set of pre-configured rules files.

To use these pre-configured rule files, create a backup of your original `/etc/audit/audit.rules` file and copy the configuration file of your choice over the `/etc/audit/audit.rules` file:

```
cp /etc/audit/audit.rules /etc/audit/audit.rules_backup
cp /usr/share/doc/audit-version/stig.rules /etc/audit/audit.rules
```

For more information on auditd process, refer to the link.



CHAPTER 6

Prometheus and Grafana

- Introduction, on page 101
- Enable Prometheus, on page 101
- Add Datasource in Grafana for Prometheus, on page 103
- Grafana, on page 104
- Configure Grafana Users using CLI, on page 104
- Connect to Grafana, on page 105
- Grafana Administrative User, on page 106
- Configure Grafana for First Use, on page 112
- Manual Dashboard Configuration using Grafana, on page 114
- Configure Useful Dashboard Panels, on page 119
- Copy Dashboards and Users to pcrfclient02, on page 121
- Configure Garbage Collector KPIs, on page 122
- Export and Import Dashboards, on page 125
- Export Graph Data to CSV, on page 126
- Session Consumption Report , on page 128

Introduction

Prometheus is an application which is a part of monitoring solution in CPS. It is used to actively gather statistics from the running virtual machines and application services.

Prometheus application resides on both pcrfclient VMs. It scrapes statistics from collectd exporter after every configured interval and stores in `/var/data/Prometheus` directory on pcrfclient VMs.

To learn more about Prometheus, refer to: https://prometheus.io/docs/introduction/overview/.

Enable Prometheus

This following sections provides information on how to enable Prometheus on CPS system.

- By default, Prometheus is disabled on system. You need to configure Prometheus to start its operation.
- You can configure Prometheus using CSV based configurations or API based configurations.

- By default, statistics granularity is set to 10 seconds. To change it, you need to configure statistics granularity. Support is present for both CSV/API based installations.
- After enabling Prometheus, you must add Prometheus data source in Grafana.
- When Prometheus is enabled on the system, existing dashboards created with graphite will not work. You must use Prometheus queries to create new dashboard on the system.

CSV Based Installation Configuration Parameters

Table 12: CSV Based Installation Parameters

Parameter	Description
enable_prometheus	This parameter is used to enable/disable Prometheus in CPS. Default: disabled Possible Values: enabled, disabled
stats_granularity	This parameter is used to configure statistics granularity in seconds. Default: 10 seconds Possible Values: Positive Number

For example, in case of CSV based installations, you can configure **Configuration.csv** with the following parameters to enable Prometheus on Cluster Manager:

```
cat /var/qps/config/deploy/csv/Configuration.csv | tail -5
db_authentication_admin_passwd,72261348A44594381D2E84ADDD1E6D9A,
db_authentication_passwd_encryption,true,
db_authentication_readonly_passwd,72261348A44594381D2E84ADDD1E6D9A,
enable_prometheus,enabled,
stats_granularity,1,
```

After configuring the parameters, run the following commands to import the new configuration to VMs:

```
/var/qps/install/current/scripts/import/import_deploy.sh
/var/qps/install/current/scripts/upgrade/reinit.sh
```

API Based Installation Parameters

Table 13: API Based Installation Parameters

Parameter	Description
enablePrometheus	This parameter is used to enable/disable Prometheus in CPS. Default: disabled Possible Values: enabled, disabled
statsGranularity	This parameter is used to configure statistics granularity in seconds. Default: 10 seconds Possible Values: Positive Number

In case of API based installations, you need to use `api/system/config/config` PATCH API from Cluster Manager.

For example:

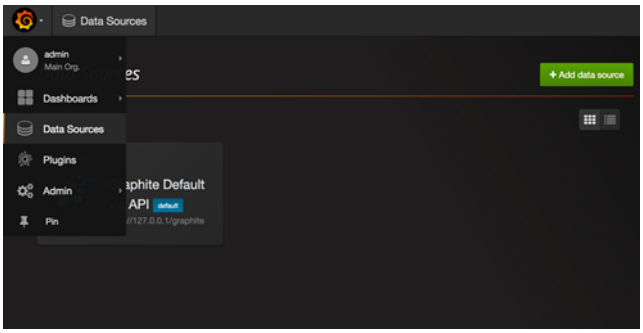
```
cat prom.yaml
 enablePrometheus: "enabled"
 statsGranularity: "1"
curl -i -X PATCH http://installer:8458/api/system/config/config -H "Content-Type:
application/yaml" --data-binary @prom.yaml
 HTTP/1.1 200 OK
  Date: Fri, 20 Apr 2018 08:38:20 GMT
  Content-Length: 0
```

Add Datasource in Grafana for Prometheus

Step 1 Login to Grafana with admin credentials.

Step 2 Click on the Grafana logo to open the sidebar menu.

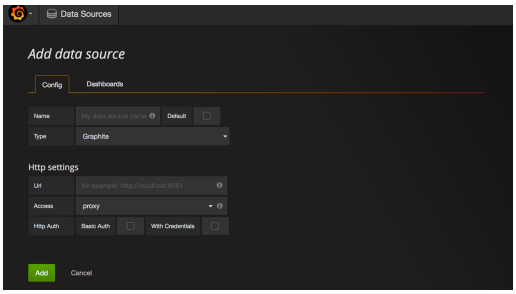
Figure 17: Sidebar Menu



Step 3 Click on **Data Sources** in the sidebar.

Step 4 Click on **Add data source**.

Figure 18: Add data source



Step 5 From **Type** drop-down list, select **Prometheus**.

Step 6 Set the appropriate Prometheus server URL (for example, http://localhost:9090/).

Step 7 Click **Add** to save the new data source.

Step 8 Create graph with Prometheus as a data source.

For example, sample graph which gives 1 min load average of VMs.

Figure 19: Sample Graph

Grafana

Grafana is a third-party metrics dashboard and graph editor.

Grafana provides a graphical or text-based representation of statistics and counters collected in the Prometheus database. To use Prometheus in Grafana, refer to http://docs.grafana.org/features/datasources/prometheus/.

Additional Grafana Documentation

This chapter provides information about the CPS implementation of Grafana. For more information about Grafana, or access the general Grafana documentation, refer to: http://docs.grafana.org.

Configure Grafana Users using CLI

In CPS 7.0.5 and higher releases, users must be authenticated to access Grafana. No default users are provided. In order to access Grafana, you must add at least one user as described in the following sections.

The steps mentioned in the sections describe how to add and delete users who are allowed view-only access of Grafana. In order to create or modify dashboards, refer to Grafana Administrative User, on page 106.

After adding or deleting a Grafana user, manually copy the `/var/broadhop/.htpasswd` file from the pcrfclient01 VM to the pcrfclient02 VM.

Also, run `/var/qps/bin/support/grafana_sync.sh` to synchronize the information between two OAM (pcrfclient) VMs.

There is no method to change the password for a Grafana user; you can only add and delete users. The `change_passwd.sh` script cannot be used to change the password for Grafana users.

Log on to the pcrfclient01 VM to perform any of the following operations.

Add First User

Step 1 Run the following command on the pcrfclient01 VM to create first user and encrypt the password:

```
/usr/bin/htpasswd -cs /var/broadhop/.htpasswd user1
```

Step 2 When prompted for a password, enter and re-enter the password.

This step creates a password file and forces SHA encryption of the password.

Add Another User

Step 1 Run the following command on the pcrfclient01 VM to create another user:

```
/usr/bin/htpasswd -s /var/broadhop/.htpasswd user2
```

Step 2 When prompted for a password, enter and re-enter the password.

This step creates a password file and forces SHA encryption of the password.

Delete a User

Run the following command on the pcrfclient01 VM:

```
/usr/bin/htpasswd -D /var/broadhop/.htpasswd user2
```

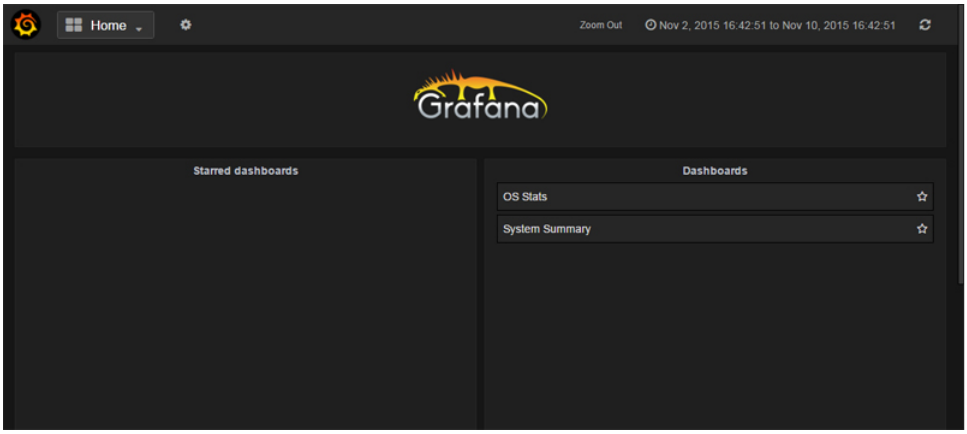
Connect to Grafana

Use the following URL to access Grafana.

- HA: https://<*lbvip01*>:9443/grafana
- All in One: http://<*ip*>:80/grafana

When prompted, enter the username and password of a user you created in Configure Grafana Users using CLI, on page 104.

Figure 20: Grafana Home Screen



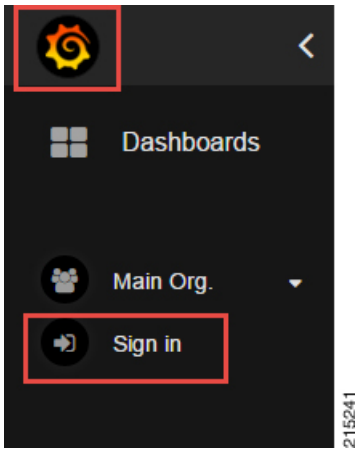
Grafana Administrative User

Log in as Grafana Admin User

To create or modify dashboards in Grafana, you must log in as the Grafana administrative user.

Step 1 Click the Grafana logo in the upper left corner of your screen.

Figure 21: Grafana Logo



Step 2 Click **Sign In**.

Step 3 Enter the administrative username and password: `admin/admin`

Change Grafana Admin User Credentials

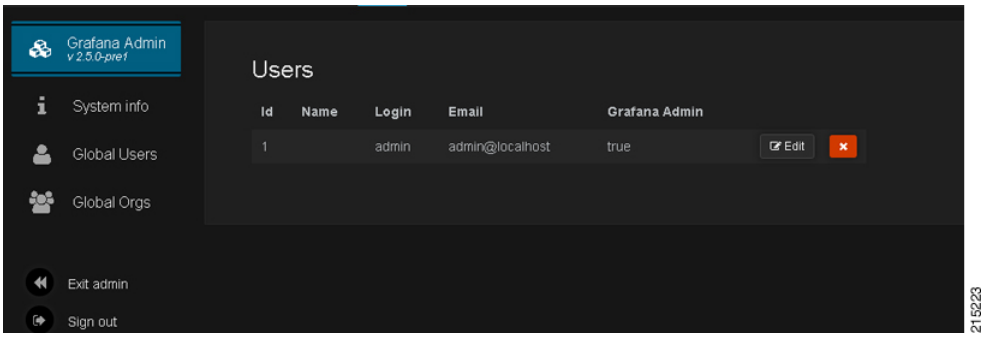
Step 1 Log in as the administrative user (`admin`/`admin`).

Step 2 Click the Grafana logo, then click **Grafana admin**.

Step 3 Click **Global Users**.

Step 4 Click **Edit**.

Figure 22: Changing Grafana Admin User Credentials



Add a Grafana User

Note The steps mentioned here can be performed only by **administrative** user.

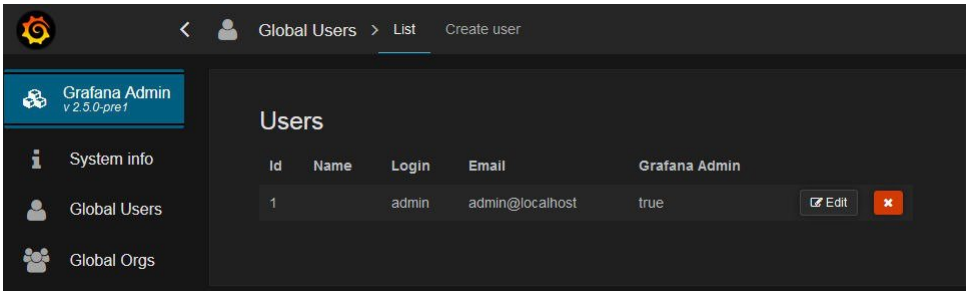
Step 1 Click the Grafana logo in the upper left corner of your screen.

Step 2 Click **Sign in**. Enter the administrative username and password.

Step 3 Click **Grafana admin** from the left side to open the **System info** pane on the right side.

Step 4 Click **Global Users** to open a pane. By default, the **List** tab appears displaying the list of users currently configured in Grafana.

Figure 23: List Tab



Step 5 Click **Create user** at the top to open **Create a new user** pane.

Figure 24: Create a new user

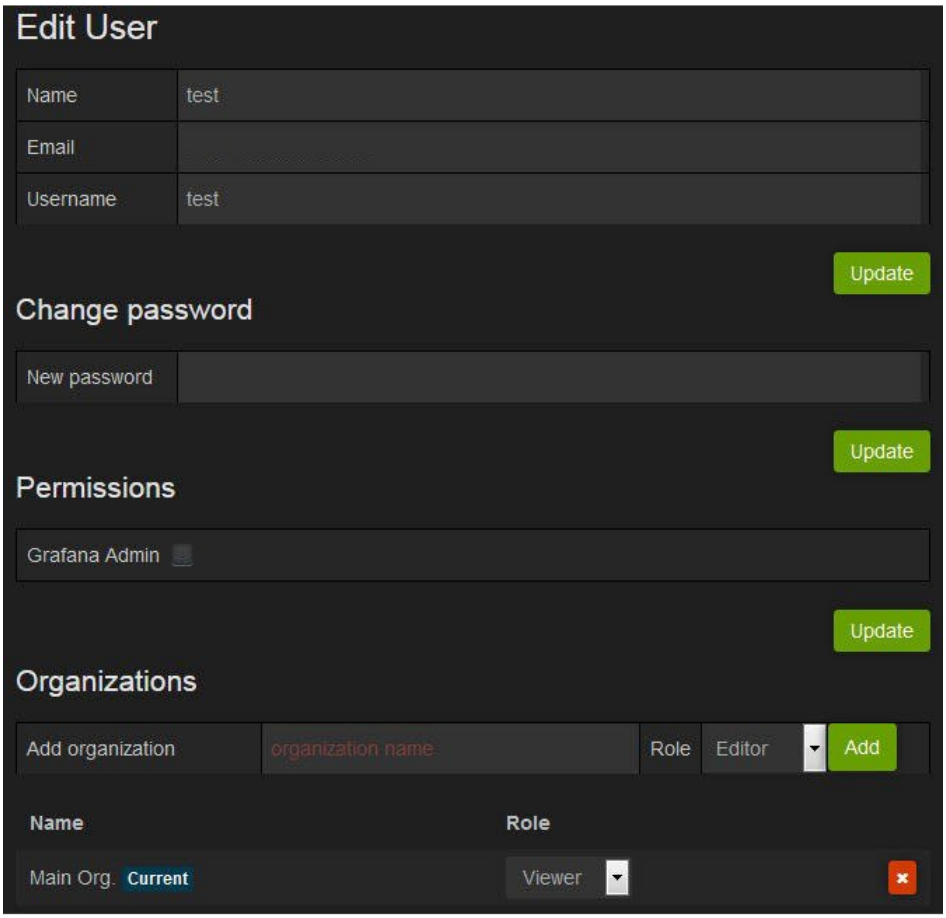
Step 6 Enter the required parameters in *Name*, *Email*, *Username* and *Password* fields.

Step 7 Click **Create** to create the grafana user.

Step 8 You will see the newly added user in the **List** tab. By default, the new user will have only **Viewer** rights.

Step 9 Click **Edit** to open **Edit User** pane. Only administrative user can update/modify the user properties.

Figure 25: Edit User Information



Change the Role of Grafana User

You can also change the rights of the user from the main page.

Note The steps mentioned here can be performed only by **administrative** user.

Click **Main Org.** drop-down list to select **Users**. This will open **Organization users** pane, where you can change the role of a user from **Role** drop-down list.

The user can have Admin/Viewer/Editor/Read Only Editor roles.

- **Admin:** An admin user can view, update and create dashboards. Also the admin can edit and add data sources and organization users.
- **Viewer:** A viewer can only view dashboards, not save or create them.

- **Editor:** An editor can view, update and create dashboards.
- **Read Only Editor:** This role behaves just like the Viewer role. The only difference is that you can edit graphs and queries but not save dashboards. The Viewer role has been modified in Grafana 2.1 so that users assigned this role can no longer edit panels.

Add an Organization

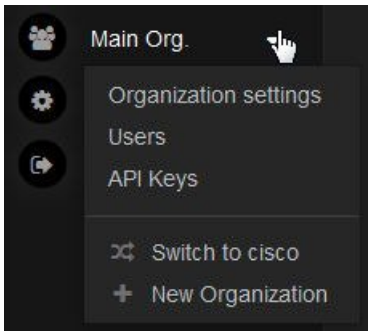
Grafana supports multiple organizations in order to support a wide variety of deployment models, including using a single Grafana instance to provide service to multiple potentially untrusted Organizations.

In many cases, Grafana will be deployed with a single Organization. Each Organization can have one or more Data Sources. All Dashboards are owned by a particular Organization.

Note The steps mentioned here can be performed only by **administrative** user.

Step 1 Click **Main Org.** drop-down list to select **New Organization**.

Figure 26: New Organization



Step 2 This will open a new pane **Add Organization**. Enter organization name in *Org. name* field. For example, test.

Step 3 After adding the name, click **Create** to open **Organization** pane.

Figure 27: Organization

In this pane, you can modify the organization name and other organization information. After modifying the information, click **Update** to update the information.

Move Grafana User to another Organization

Note The steps mentioned here can be performed only by **administrative** user.

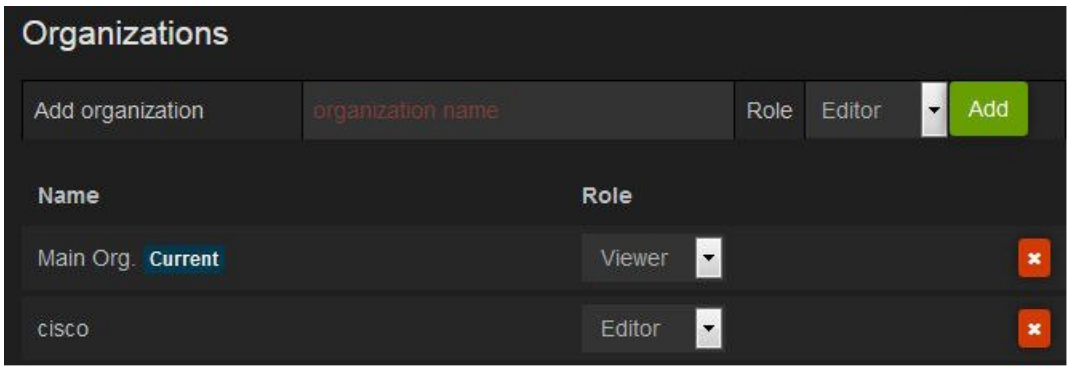
Step 1 Click **Grafana admin** from the main page to **System Info** page.

Step 2 Click **Global Users** from the left pane to open **Users** pane on the right.

Step 3 Click **Edit** against the user for whom you want to make the changes.

Step 4 Under **Organizations** section, you can add the user to some other organizations.

Figure 28: Move User to another Organization



Step 5 In *Add organization* field, you need to enter the name of the new organization.

Step 6 You can also change the role of the user from the **Role** drop-down list.

Step 7 After adding the required information, click **Add** to add the user into a new organization.

Step 8 In the above example, you can see that the user is added to the new organization. If you want to remove the user from pervious organization, click the **red cross** at the end.

Configure Grafana for First Use

After an initial installation or after upgrading an existing CPS deployment which used Grafana, you must perform the steps in the following sections to validate the existing data sources.

Migrate Existing Grafana Dashboards

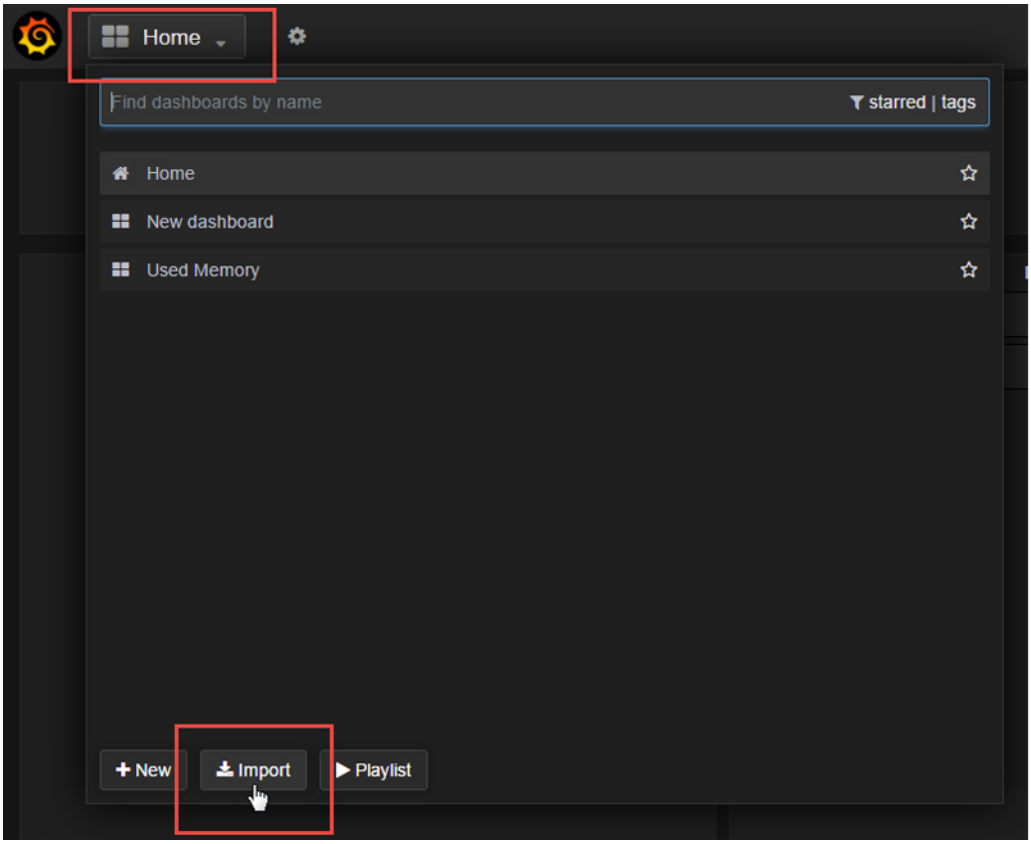
During an upgrade of CPS (and Grafana), saved dashboard templates remain intact.

After upgrading an existing CPS deployment, you must manually migrate any existing Grafana dashboards.

Step 1 Sign in as the Grafana Administrative User. For more information, refer to Grafana Administrative User, on page 106.

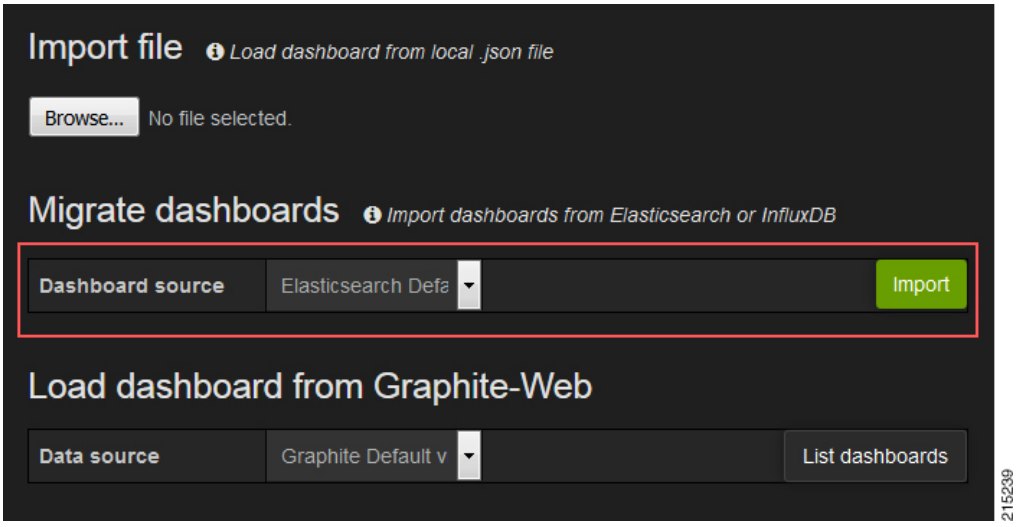
Step 2 Click **Home** at the top of the Grafana window and then click **Import** as shown below:

Figure 29: Import



Step 3 In the Migrate dashboards section, verify that **Elasticsearch Def** (Elasticsearch Default via API) is listed, then click **Import**.

Figure 30: Import File



Step 4 All existing dashboards are imported and should now be available.

Manual Dashboard Configuration using Grafana

Grafana enables you to create custom dashboards which provide graphical representations of data by fetching information from the Prometheus database. Each dashboard is made up of panels spread across the screen in rows.

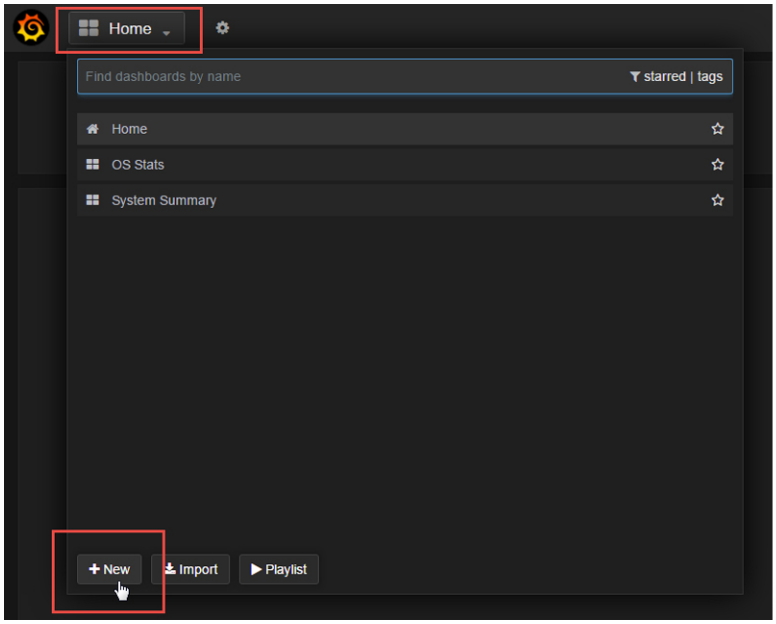
Note CPS includes a series of preconfigured dashboard templates. To use these dashboards, refer to Updating Imported Templates.

Create a New Dashboard Manually

Step 1 Sign-in as a Grafana Administrative user. For more information, see Grafana Administrative User, on page 106.

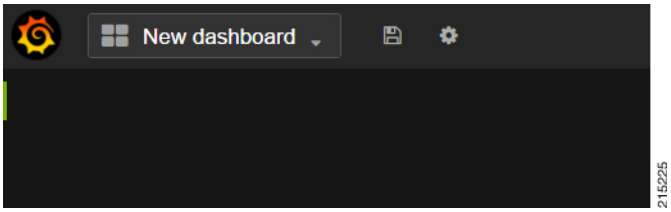
Step 2 Click **Home** at the top of the Grafana window and select **New** as shown below:

Figure 31: Home



A blank dashboard is created.

Figure 32: Blank Dashboard



Step 3 At the top of the screen, click the gear icon, then click **Settings**.

Figure 33: Gear Icon

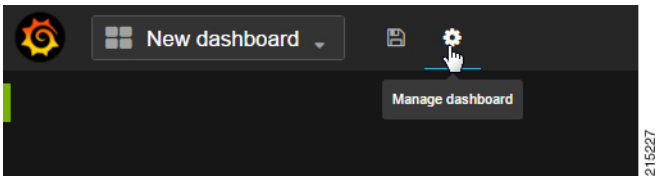
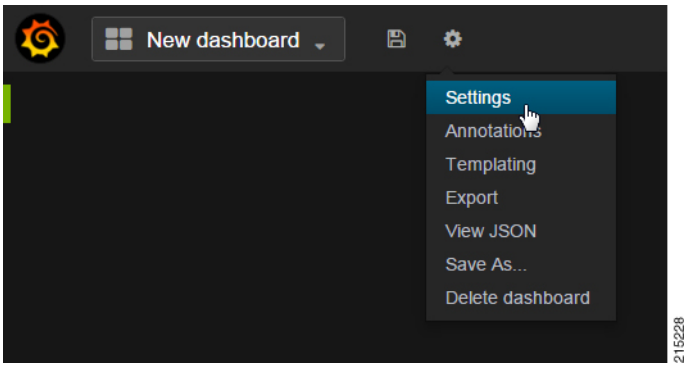
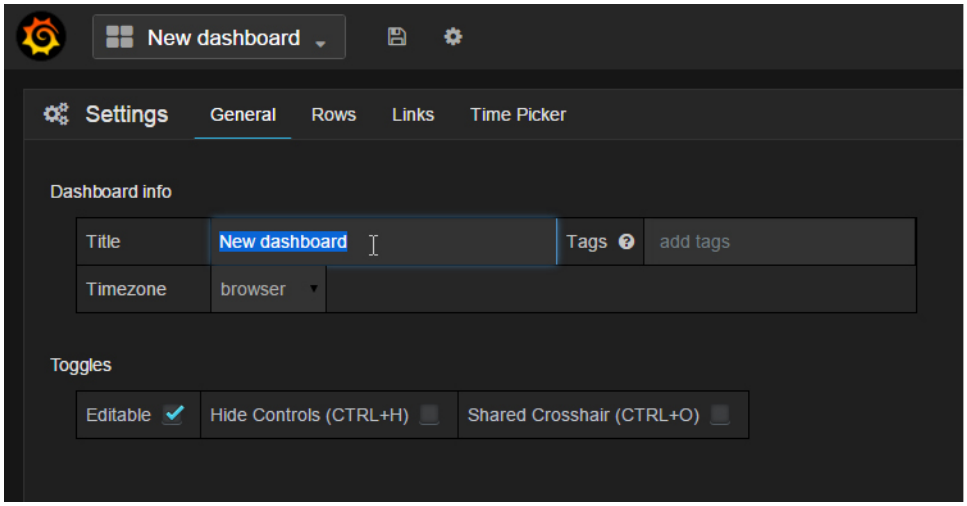


Figure 34: Settings



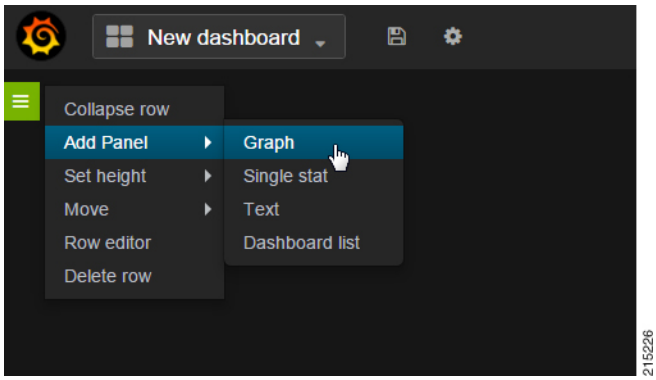
Step 4 Provide a name for the dashboard and configure any other Dashboard settings. When you have finished, click the **X** icon in the upper right corner to close the setting screen.

Figure 35: Name for Dashboard



Step 5 To add a graph to this dashboard, hover over the green box on the left side of the dashboard, then point to **Add Panel**, then click **Graph**.

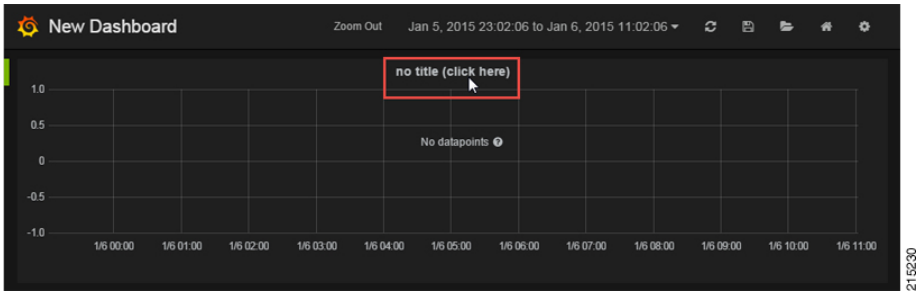
Figure 36: Add Graph to Dashboard



Configure Data Points for the Panel

Step 1 Click the panel title, as shown below, then select **Edit**.

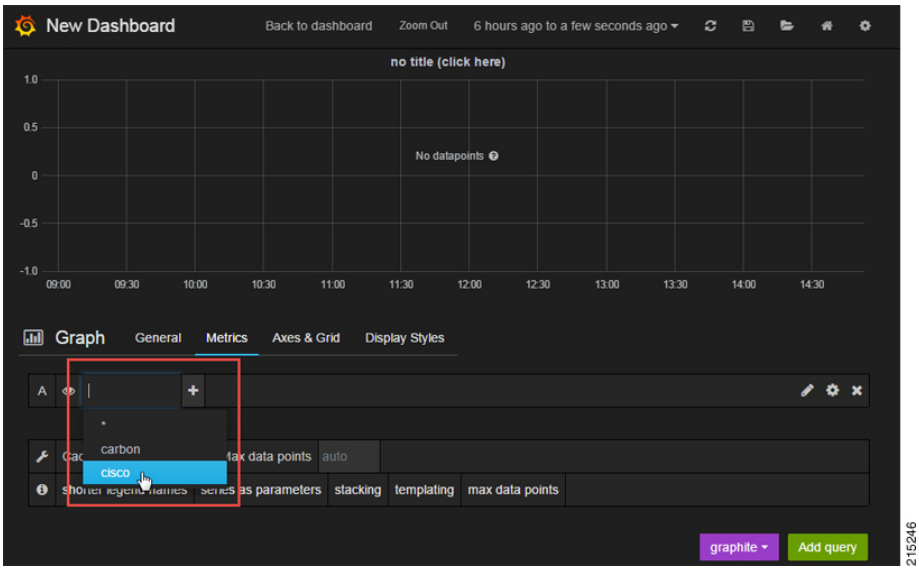
Figure 37: Edit



Step 2 Select the necessary metrics by clicking on the select metric option provided in the query window. A drop-down list appears from which you can choose the required metrics.

Select metrics by clicking select metric repeatedly until the lowest level of the hierarchy.

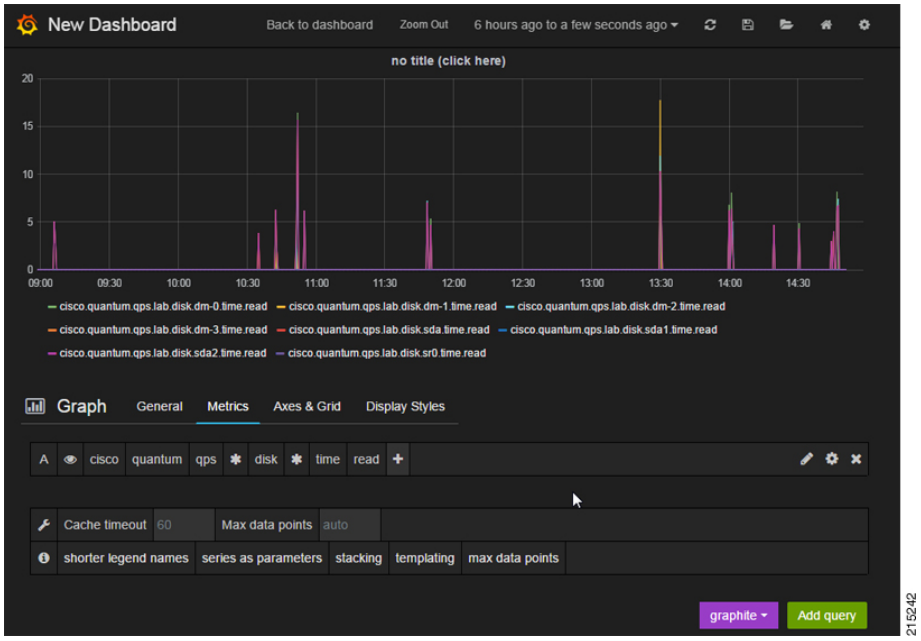
Figure 38: Metric Selection



Note Clicking the '*' option in the drop-down list selects all the available metrics.

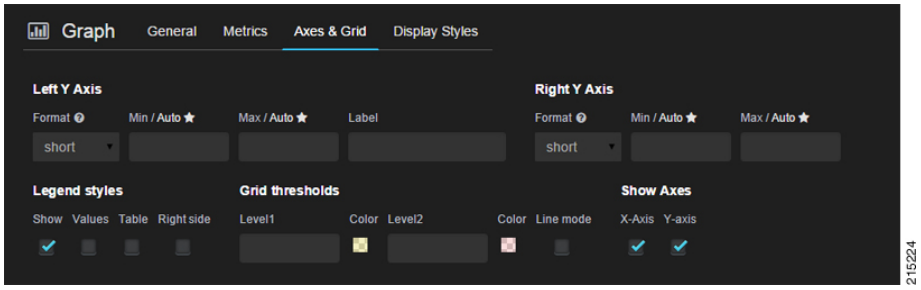
Step 3 Click the '+' tab to add aggregation functions for the selected metrics. the monitoring graph is displayed as shown below.

Figure 39: Aggregation Functions



Step 4 The x-axis and y-axis values can be configured in the **Axes & Grid** tab.

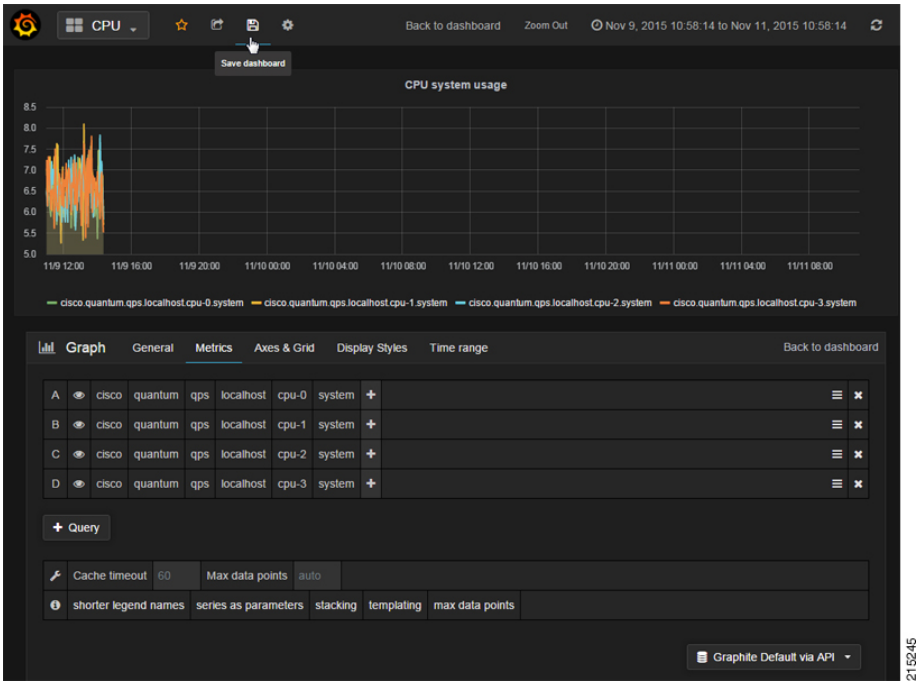
Figure 40: Axes and Grid



Step 5 Click the disk icon (Save dashboard) at the top of the screen, as shown in the following image.

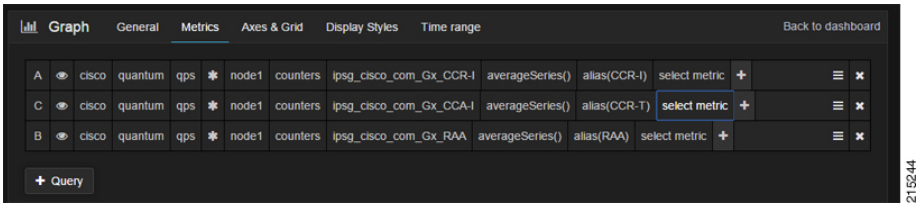
Note The changes to this dashboard are lost if you do not click the **Save** icon.

Figure 41: Save



Graphical representation of application-messages such as - CCR, CCA, Gx, Gy, LDAP, Rx messages and so on, can be configured in the dashboard panel by using the queries shown in the below figure.

Figure 42: Graphical Representation



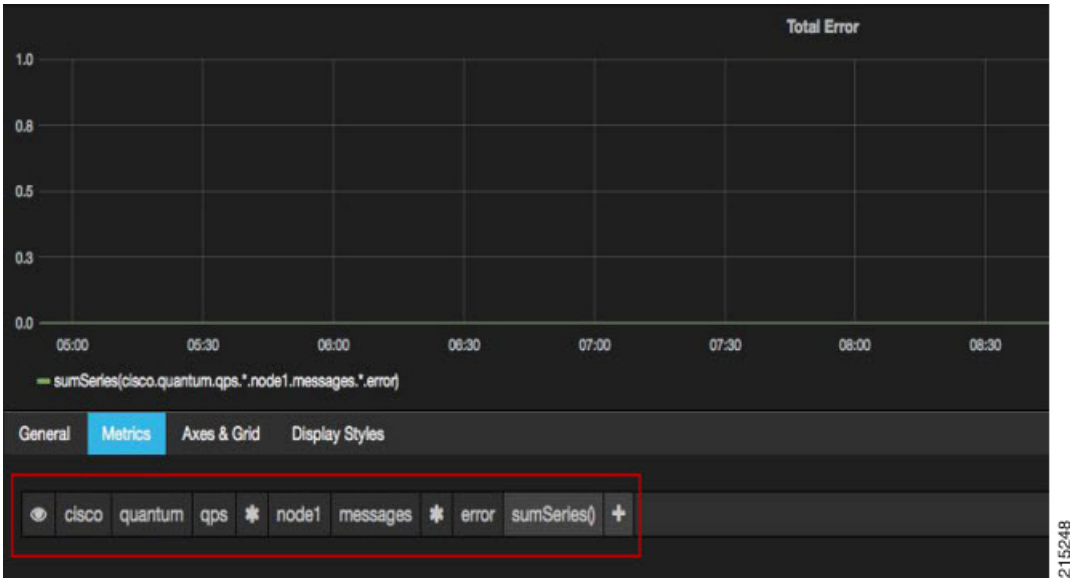
Configure Useful Dashboard Panels

The following section describes the configuration of several useful dashboard panels that can be used while processing Application Messages. Configure the dashboard panel as shown in the screens below.

Total Error:

This dashboard panel lists the errors found during the processing of Application Messages. To configure Total Error dashboard panel, create a panel with name 'Total Error' and configure its query as shown:

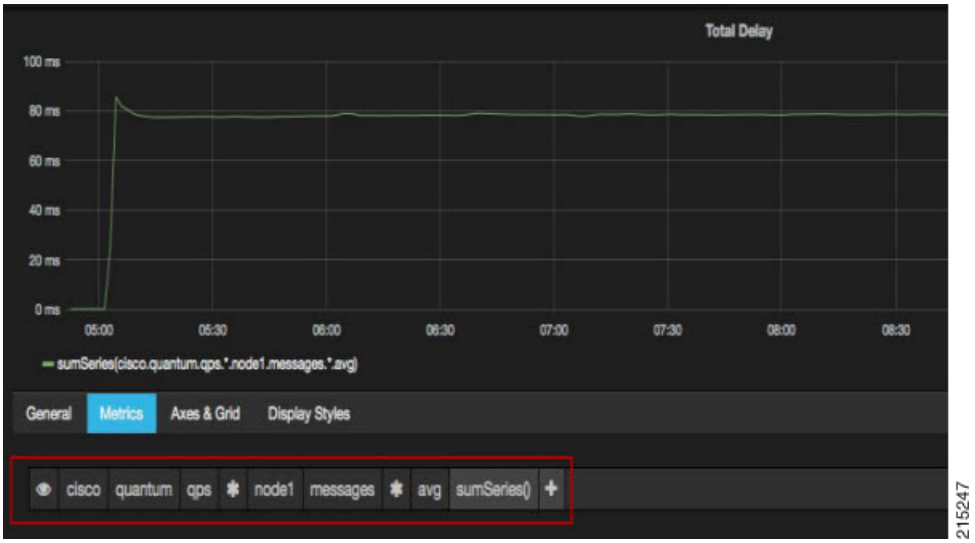
Figure 43: Total Error Dashboard



Total Delay:

This dashboard panel displays the total delay in processing various Application Messages. To configure Total Delay dashboard panel, create a panel with name Total Delay and configure its query as shown:

Figure 44: Total Delay Dashboard



Total TPS:

This panel displays the total TPS of CPS system. Total TPS count includes all Gx, Gy, Rx, Sy, LDAP and so on. The panel can be configured as shown below:

Figure 45: Total TPS

Updating Imported Templates

Some of the preconfigured templates (such as Diameter statistics panels) have matrices configured which are specific to a particular set of Diameter realms. These panels need to be reconfigured to match customer specific Diameter realms.

For example, the Gx P-GW panel in the Diameter Statistics dashboard does not fetch the stats and displays the message “No Datapoints”. The probable reasons could be:

- Matrices used in query uses matrices specific to particular Diameter realm which is different on customer setup.
- No application call of such type has ever landed on CPS Policy Directors (LBs) (no Diameter call from the P-GW has ever landed on Policy Director after the Grafana setup).

Copy Dashboards and Users to pcrfclient02

As a best practice, the internal Grafana database should be kept in sync between pcrfclient01 and pcrfclient02. This sync operation should be performed after any dashboard or Grafana user is migrated, updated, added or removed.

Under normal operating conditions, all Grafana operations occur from pcrfclient01. In the event of a pcrfclient01 failure, pcrfclient02 is used as backup, so keeping the database in sync provides a seamless user experience during a failover.

The following steps copy all configured Grafana dashboards, Grafana data sources, and Grafana users configured on pcrfclient01 to pcrfclient02.

Log in to the pcrfclient01 VM and run the following command:

```
/var/qps/bin/support/grafana_sync.sh
```

As a precaution, the existing database on pcrfclient02 is saved as a backup in the `/var/lib/grafana` directory.

Configure Garbage Collector KPIs

The following sections describe the steps to configure Garbage Collector (GC) KPIs in Grafana:

- Backend changes: Changes in the collectd configuration so that GC related KPIs will be collected by collectd and stored in graphite database.
- Frontend changes: Changes in Grafana GUI for configuring metrics for GC graph.

Backend Changes

Check if the following changes are already present in the jmxplugin.conf file. If already configured, then skip this section and move to configuring the Grafana dashboard.

Step 1 Edit `/etc/puppet/modules/qps/templates/collectd_worker/collectd.d/jmxplugin.conf` on the Cluster Manager VM as described in the following steps.

Step 2 Verify that the JMX plugin is enabled. The following lines must be present in the `jmxplugin.conf` file.

```
JVMARG has path for jmx jar

JVMARG

-Djava.class.path=/usr/share/collectd/java/collectd-api.jar/usr/share/collectd/java/generic-jmx.jar

And GenericJMX plugin is loaded

LoadPlugin org.collectd.java.GenericJMX
```

Step 3 Add an Mbean entry for garbage collector mbean in GenericJMX plugin so that statistics from this mbean will be collected.

```
  # Garbage collector information
<MBean "garbage_collector">
      ObjectName "java.lang:type=GarbageCollector,*"
      InstancePrefix "gc-"
      InstanceFrom "name"
<Value>
        Type "invocations"
        #InstancePrefix ""
        #InstanceFrom ""
        Table false
        Attribute "CollectionCount"
</Value>
<Value>
        Type "total_time_in_ms"
        InstancePrefix "collection_time"
        #InstanceFrom ""
        Table false
        Attribute "CollectionTime"
</Value>
</MBean>
```

Step 4 For every “Connection” block in `jmxplugin.conf` file add the entry for garbage collector mbean.

For example:

```
<Connection>
  InstancePrefix "node1."
ServiceURL "service:jmx:rmi:///jndi/rmi://localhost:9053/jmxrmi"
Collect "garbage_collector"
Collect "java-memory"
Collect "thread"
Collect "classes"
Collect "qns-counters"
Collect "qns-actions"
Collect "qns-messages"
</Connection>]
```

Step 5 Save the changes to the `jmxplugin.conf` file then synchronize the changes to all CPS VMs as follows:

a) Go to the `/var/qps/install/current/scripts/build/` directory on the Cluster Manager and execute the following script:

```
./build_puppet.sh
```

b) Go to the `/var/qps/install/current/scripts/upgrade/` directory on the Cluster Manager and execute the following command:

```
./reinit.sh
```

c) Restart the collectd service on all VMs by running the following command on each VM in the CPS cluster:

```
monit restart collectd
```

Frontend Changes

The frontend changes must be done in the Grafana GUI.

Step 1 Create a new Grafana dashboard. For more information, see Manual Dashboard Configuration using Grafana, on page 114.

Step 2 In the **Metrics** tab of the new dashboard, configure queries for GC related KPIs.

The query needs to be configured in the following format:

```
cisco.quantum.qps.<hostname>.node*. gc*.total_time_in_ms-collection_time

cisco.quantum.qps.<hostname>.node*.gc*.invocations
```

where, <*hostname*> is regular expression for the name of hosts from which KPI needs to be reported.

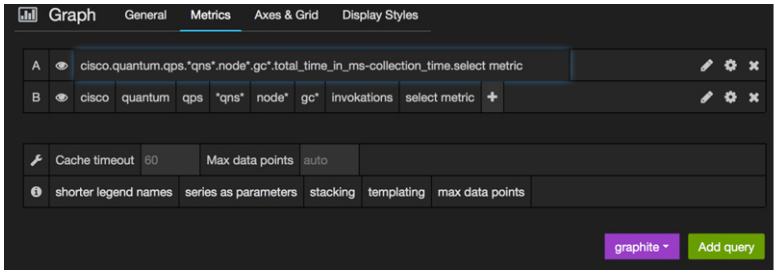
If this is a CPS All in One (AIO) deployment, the hostname is “lab”.

If this is a High Availability (HA) CPS deployment, KPIs need to be reported from all Policy Server (QNS) VMs.

Assuming the Policy Server (QNS) VMs have “qns” in their hostname, then a regular expression would be *qns*. This would report data for all VMs that have a hostname containing “qns” (qns01 qns02 and so on).

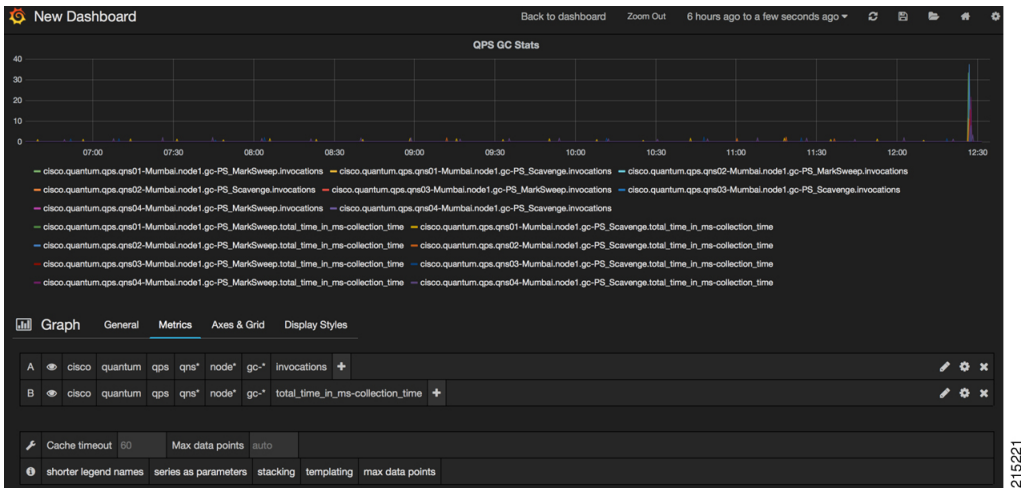
- AIO Setup

Figure 46: On AIO Setup



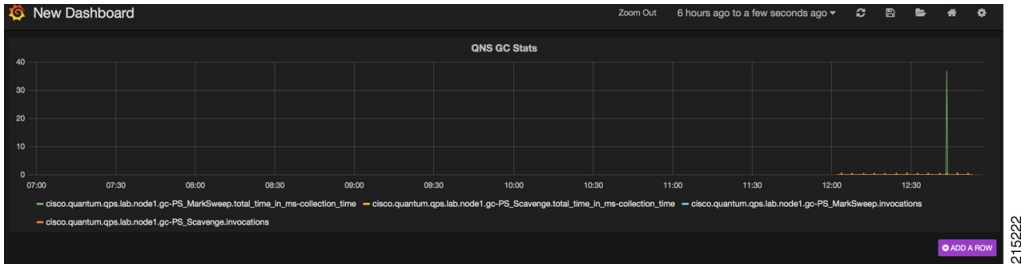
- HA Setup

Figure 47: On HA Setup



An example statistics graph is shown below.

Figure 48: Example Graph



Step 3 Save the dashboard by clicking on Save icon.

Export and Import Dashboards

Existing dashboard templates can be exported and imported between environments. This is useful for sharing Grafana dashboards with others.

Export Dashboard

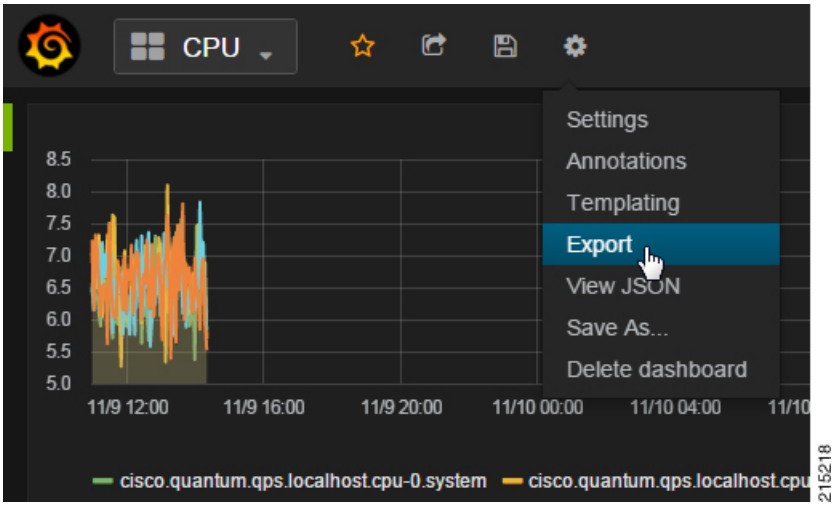
This topic describes how to export a dashboard configuration to a file.

Step 1 Sign-in as a Grafana Administrative User.

Step 2 Open the dashboard to be exported.

Step 3 Click the gear icon at the top of the page, and then select **Export** to save the dashboard configuration on your local system.

Figure 49: Export



Step 4 If prompted, select the location on your local system to save the dashboard template, and click **OK**.

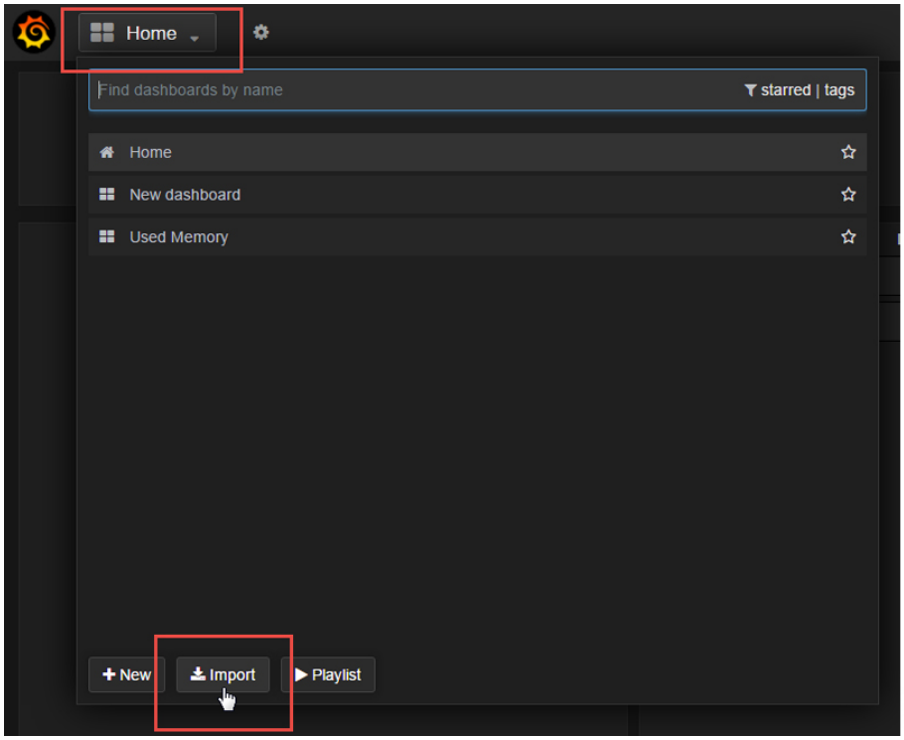
Import Dashboard

This topic describes how to import a dashboard from a file.

Step 1 Sign-in as a Grafana Administrative User.

Step 2 Click **Home** at the top of the Grafana window, and then click **Import** as shown below.

Figure 50: Import



Step 3 Click **Choose File**.

Step 4 Select the file on your local system to save the dashboard template and click **Open**.

Step 5 After the dashboard is loaded, click the disk icon (Save dashboard) at the top of the screen to save the dashboard.

Note Your changes to this dashboard are lost if you do not save the dashboard.

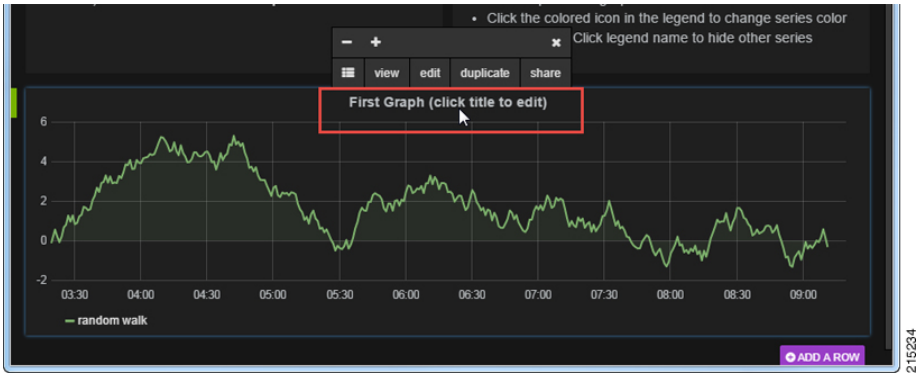
The data to be imported in the dashboard should be in the correct format. Grafana does not throw any error if incorrectly formatted data is loaded.

Export Graph Data to CSV

This topic describes how to export the data in a graph panel to a CSV file.

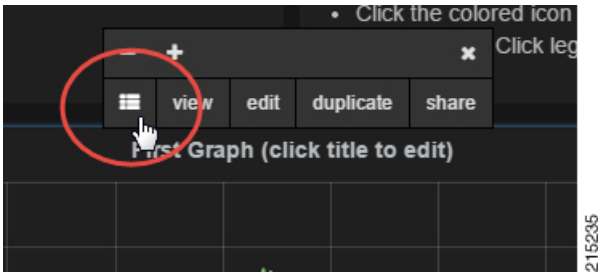
Step 1 Click the title of the graph as shown below to open the graph controls.

Figure 51: Title



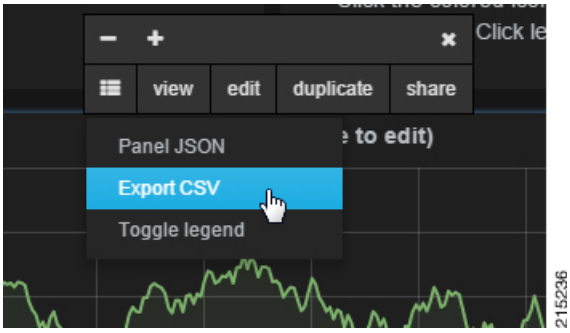
Step 2 Click the rows button to open another menu.

Figure 52: Rows



Step 3 Click **Export CSV**.

Figure 53: Export



A `grafana_data_export.csv` file is downloaded by your browser.

Session Consumption Report

Introduction

This feature generates the session consumption report and stores the data into a separate log. The total number of sessions limited by the license, the total number of active sessions, and total transactions per second are documented at regular time intervals into the log. The core license number is derived from the license file that has the total number of sessions limited by the license. The active session count and the transaction count has been taken from Grafana using the graphite query. A single entity of the feature mainly prints the current time stamp with the statistics values.

Data Collection

The session and TPS count is collected from the graphite API with a JSON response. The JSON response is then parsed to get the counter, which is then logged into the consolidated log. The sample URL and the JSON response are given below:

```
> curl
 "http://localhost:8008/render?target=cisco.quantum.qps.pcrfclient01.set_session_count_
total.records&from=-20second&until=-0hour&format=json"
> [{"target":
"cisco.quantum.qps.localhost.set_session_count_total.records", "datapoints": [[3735.42,
1455148210], [3748.0, 1455148220]]}]
> curl
 "http://localhost:8008/render?target=sumSeries(cisco.quantum.*.*.node*.messages.e2e*.success)
&from=-20second&until=-0hour&format=json"
>  [{"target":
 "sumSeries(cisco.quantum.*.*.node*.messages.e2e*.success)", "datapoints": [[2345.34324,
1455148210], [2453.23445453,
1455148220]]}]
```

Logging

Data logging is done using the logback mechanism. The consolidated data that is generated is stored in a separate log file named `consolidated-sessions.log` inside the `/var/log/broadhop` directory along with other logs. The data entries are appended to the log every 90 seconds. The logs generated are detailed and have the counter name and the current value with the time stamp.

Performance

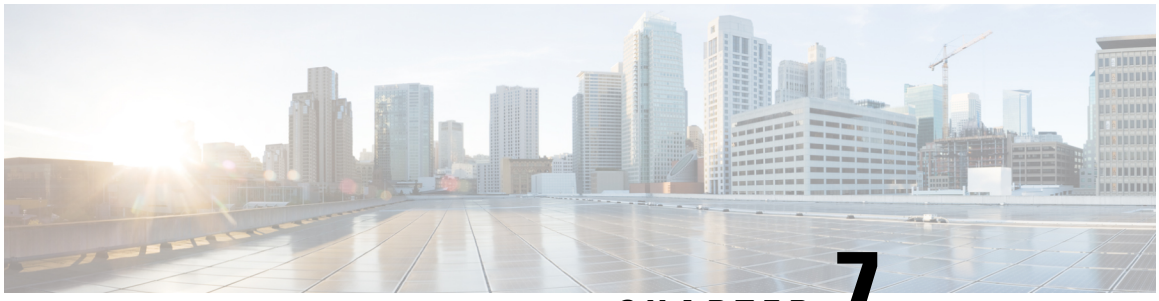
The codebase pulls the JSON response from the Graphite API. The overhead by the codebase adds an average of 350 ms of time.

Log Rotation

A log rotation policy is applied on the logs generated for the session Consumption Report. The file size limitation for each log file is 100 MB. The limitation on number of log files is 5. The logs get rotated after reaching the limitations. One file contains a little more than two years of data, so five such files can contain 10 years of data until the first file get replaced.

Sample Report

```
2016-02-15 20:30:01 - TPS_COUNT: 6440.497603         SESSION_COUNT: 200033.0
LICENSE_COUNT: 10000000
2016-02-15 20:31:31 - TPS_COUNT: 6428.235699999999   SESSION_COUNT: 201814.0
LICENSE_COUNT: 10000000
2016-02-15 20:33:01 - TPS_COUNT: 5838.386624000001   SESSION_COUNT: 204818.0
LICENSE_COUNT: 10000000
2016-02-15 20:34:31 - TPS_COUNT: 6266.777699999999   SESSION_COUNT: 208719.0
LICENSE_COUNT: 10000000
2016-02-15 20:36:01 - TPS_COUNT: 6001.863687         SESSION_COUNT: 211663.0
LICENSE_COUNT: 10000000
2016-02-15 20:37:31 - TPS_COUNT: 6528.9450540000025  SESSION_COUNT: 213976.0
LICENSE_COUNT: 10000000
2016-02-15 20:39:01 - TPS_COUNT: 6384.073428         SESSION_COUNT: 218851.0
LICENSE_COUNT: 10000000
2016-02-15 20:40:31 - TPS_COUNT: 6376.373494000002   SESSION_COUNT: 220515.0
LICENSE_COUNT: 10000000
2016-02-15 20:42:01 - TPS_COUNT: 6376.063389999998   SESSION_COUNT: 222308.0
LICENSE_COUNT: 10000000
2016-02-15 20:43:31 - TPS_COUNT: 6419.310694000001   SESSION_COUNT: 223146.0
LICENSE_COUNT: 10000000
2016-02-15 20:45:01 - TPS_COUNT: 6455.804928         SESSION_COUNT: 222546.0
LICENSE_COUNT: 10000000
2016-02-15 20:46:31 - TPS_COUNT: 6200.357029999999   SESSION_COUNT: 223786.0
LICENSE_COUNT: 10000000
2016-02-15 20:48:02 - TPS_COUNT: 6299.090987         SESSION_COUNT: 223973.0
LICENSE_COUNT: 10000000
2016-02-15 20:49:31 - TPS_COUNT: 6294.876452         SESSION_COUNT: 226629.0
LICENSE_COUNT: 10000000
2016-02-15 20:51:01 - TPS_COUNT: 6090.202965999999   SESSION_COUNT: 227581.0
LICENSE_COUNT: 10000000
2016-02-15 20:52:31 - TPS_COUNT: 6523.586347999997   SESSION_COUNT: 228450.0
LICENSE_COUNT: 10000000
2016-02-15 20:54:01 - TPS_COUNT: 5842.613997000001   SESSION_COUNT: 229334.0
LICENSE_COUNT: 10000000
2016-02-15 20:55:31 - TPS_COUNT: 6638.526543         SESSION_COUNT: 232683.0
LICENSE_COUNT: 10000000
2016-02-15 20:57:01 - TPS_COUNT: 6073.7797439999995  SESSION_COUNT: 230466.0
LICENSE_COUNT: 10000000
2016-02-15 20:58:31 - TPS_COUNT: 6354.272679999999   SESSION_COUNT: 234070.0
LICENSE_COUNT: 10000000
2016-02-15 21:00:03 - TPS_COUNT: 6217.872034999999   SESSION_COUNT: 236139.0
LICENSE_COUNT: 10000000
```



CHAPTER 7

Managing High Availability in CPS

- Porting All-In-One Policy Builder Configuration to HA, on page 131
- HAProxy, on page 134
- Expanding an HA Deployment, on page 135
- Enable SSL, on page 137

Porting All-In-One Policy Builder Configuration to HA

This section describes how to port the Policy Builder configuration from an All-In-One (AIO) environment to a High Availability (HA) environment.

Prerequisites

- All the VMs were created using the deployment process.
- This procedure assumes the datastore that will be used to have the virtual disk has sufficient space to add the virtual disk.
- This procedure assumes the datastore has been mounted to the VMware ESX server, regardless of the backend NAS device (SAN or iSCSI, etc).

Porting the Policy Builder Configuration

Policy Builder configuration can be utilized between environments, however, the configuration for Systems and Policy Enforcement Points is environment-specific and should not be moved from one environment to another.

The following instructions will not overwrite the configuration specific to the environment. Please note that as the Systems tab and Policy Enforcement Points data is not moved, the HA system should have these things configured and running properly (as stated above).

The following steps describe the process to port a configuration from an AIO environment to an HA environment.

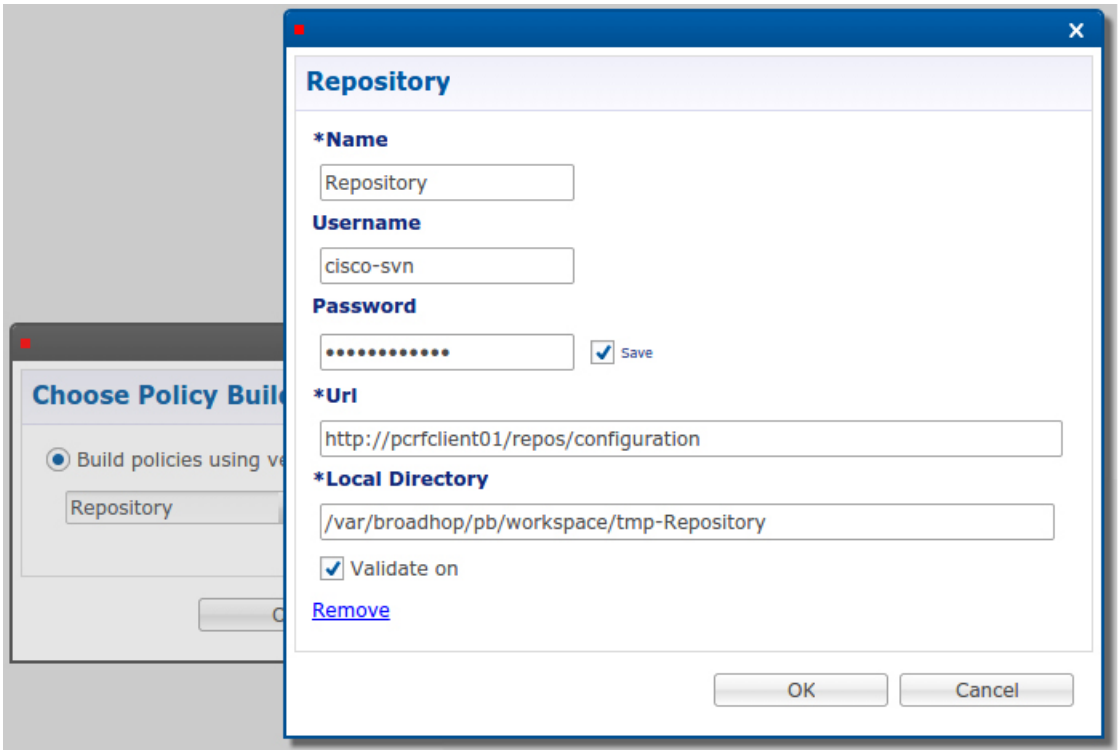
Step 1 If the HA environment is currently in use, ensure that SVN backups are up to date.

Step 2 Find the URL that Policy Builder is using to load the configuration that you want to use. You can find this by clicking **Edit** on the initial page in Policy Builder.

The URL is listed in the URL field. For the purpose of these instructions, the following URL will be used for exporting the configuration from the AIO environment and importing the configuration to the HA environment:

http://pcrfclient01/repos/configuration

Figure 54: Repository configuration



Step 3 On the AIO, export the Policy Builder configuration by entering the following commands:

```
cd /var/tmp
svn export http://pcrfclient01/repos/configuration aio_configuration
```

This creates a directory `/var/tmp/aio_configuration`.

Step 4 Remove the system configuration by entering the following commands:

```
cd aio_configuration
rm -f System* *Configuration* DiameterStack* VoucherSettings* Cluster* Instance*
```

Step 5 Move the `/var/tmp/aio_configuration` directory to `/var/tmp` on your Cluster Manager (using scp, zip and so on).

Step 6 SSH into the pcrfclient01.

The following steps assume you will replace the existing default Policy Builder configuration located at http://pcrfclient01/repos/configuration on your HA environment. If you would like to access your old configuration, copy it to a new location. For example:

```
svn cp http://pcrfclient01/repos/configuration http://pcrfclient01/repos/configuration_date
```

Then set up a new Repository in the HA Policy Builder to access the old configuration.

Step 7 Check out the old configuration (http://pcrfclient01/repos/configuration):

```
svn co http://pcrfclient01/repos/configuration /var/tmp/ha_configuration
```

Step 8 Remove the non-system configuration:

```
svn rm ls | egrep -v '(System|Configuration|DiameterStack|VoucherSettings|Cluster|Instance)'
```

Step 9 Copy in the AIO configuration files:

```
/bin/cp -f /var/tmp/aio_configuration/* .
svn add *
```

Step 10 Commit the configuration:

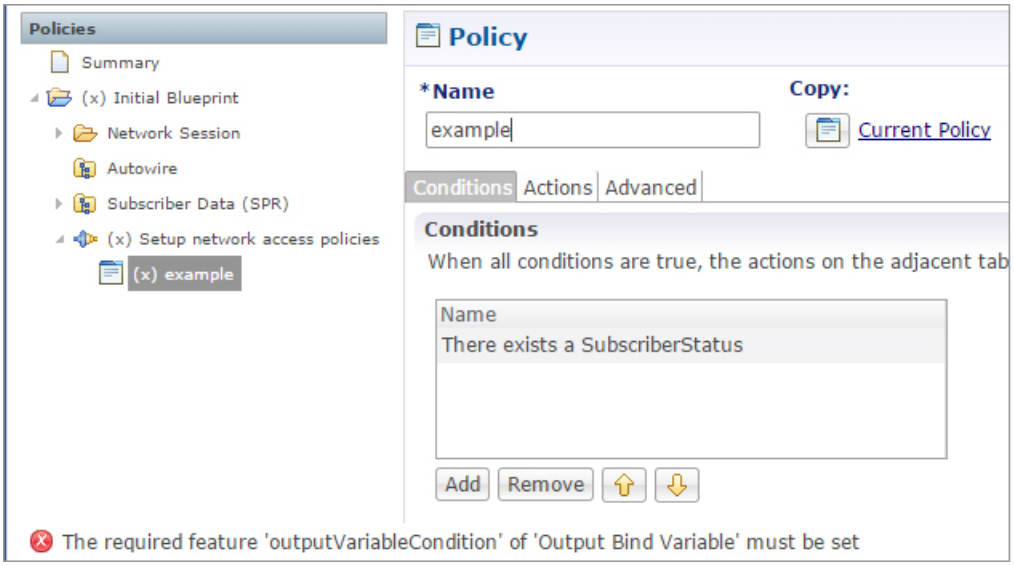
```
svn ci . -m 'commit configuration moved from AIO'
```

Step 11 If you are already logged into Policy Builder, reload the Policy Builder URL in your browser to access the new configuration.

Step 12 Check for errors in Policy Builder. This often indicates a software mismatch.

Errors are shown with an (x) next to the navigation icons in the left pane of Policy Builder. For example:

Figure 55: Error in Policy Builder



Step 13 Publish the configuration. Refer to the *CPS Mobile Configuration Guide* for detailed steps.

HAProxy

HAProxy is an opensource load balancer used in High Availability (HA) and Geographic Redundancy (GR) CPS deployments. It is used by the CPS Policy Directors (lbs) to forward IP traffic from lb01/lb02 to other CPS nodes. HAProxy runs on the active Policy Director VM.

Documentation for HAProxy is available at http://www.haproxy.org/#docs.

HAProxy Service Operations

Diagnostics

For a general diagnostics check of the HAProxy service, run the following command from any VM in the cluster (except sessionmgr):

diagnostics.sh --ha_proxy

```
QPS Diagnostics Multi-Node Environment
--------------------------
Ping Check for qns01...[PASS]
Ping Check for qns02...[PASS]
Ping Check for qns03...[PASS]
Ping Check for qns04...[PASS]
Ping Check for lb01...[PASS]
Ping Check for lb02...[PASS]
Ping Check for sessionmgr01...[PASS]
Ping Check for sessionmgr02...[PASS]
Ping Check for sessionmgr03...[PASS]
Ping Check for sessionmgr04...[PASS]
Ping Check for pcrfclient01...[PASS]
Ping Check for pcrfclient02...[PASS]
HA Multi-Node Environment
--------------------------
Checking HAProxy status...[PASS]
```

Service Commands

The following commands must be issued from the lb01 or lb02 VM.

To check the status of the HAProxy services, run the following command:

```
monit status haproxy
```

```
[root@host-lb01 ~]# service haproxy status
haproxy (pid 10005) is running...
```

To stop the HAProxy service, run the following command:

```
monit stop haproxy
```

To restart the HAProxy service, run the following command:

```
monit restart haproxy
```

HAProxy Statistics

To view statistics, open a browser and navigate to the following URL:

- **For HAProxy Statistics**: http://<diameterconfig>:5540/haproxy?stats
- **For HAProxy Diameter Statistics**: http://<diameterconfig>:5540/haproxy-diam?stats

Changing HAProxy Log Level

To change HAProxy log level in your CPS deployment, you must make changes to the HAProxy configuration files on the Cluster Manager and then push the changes out to the Policy Director (lb) VMs.

Once deployed, the HAProxy configuration files are stored locally on the Policy Director VMs at `/etc/haproxy/haproxy.cfg.erb` and `/etc/haproxy/haproxy-diameter.erb`.

Note Whenever you upgrade with latest ISO, the log level will be set to default level (err).

Step 1 Log in to the Cluster Manager.

Step 2 Create a backup of the HAProxy configuration file before continuing:

```
cp /var/qps/install/current/puppet/modules/qps/templates/etc/haproxy/haproxy.cfg.erb
/var/qps/install/current/puppet/modules/qps/templates/etc/haproxy/haproxy.cfg.erb-bak-<date>
```

Step 3 Edit the HAProxy files as needed.

By default, the logging level is set as error (err) in `/var/qps/install/current/puppet/modules/qps/templates/etc/haproxy/haproxy-diameter.erb`:

```
log         127.0.0.1       local1 err
```

By default, the logging level in `/var/qps/install/current/puppet/modules/qps/templates/etc/haproxy/haproxy.cfg.erb`:

```
log             127.0.0.1   local3 emerg alert crit err warning
```

The log level can be adjusted to any of the following log levels as needed:

emerg alert crit err warning notice info debug

Step 4 Run `build_all.sh` to rebuild the CPS VM packages.

Step 5 Run `reinit.sh` to trigger all VMs to download the latest software and configuration from the Cluster Manager.

Expanding an HA Deployment

For future installations and network upgrades, this section proposes what hardware and components you should consider as you grow your network. The CPS solution is a robust and scalable software-based solution that can be expanded by adding additional hardware and software components. The following sections explain typical scenarios of when to expand the hardware and software to effect such growth.

Typical Scenarios When Expansion is Necessary

Your network may grow for the following reasons:

- The subscriber base has grown or will grow beyond the initial installation specifications.

  In this case, the number of active or non-active subscribers becomes larger than the initial deployment. This can cause one or more components to reach capacity. New components must be added to accommodate the growth.

- The services or subscriber scenarios have changed, or new services have been introduced, and the transactions per second on a component no longer meet requirements.

  When a new service or scenario occurs, often there is a change in the overall Transactions Per Second (TPS), or in the TPS on a specific component. When this occurs, new components are necessary to handle the new load.

- The operator notices that there are factors outside of the initial design that are causing either the overall system or a specific component to have a high resource load.

  This may cause one or multiple components to reach its capacity for TPS. When this occurs, new components are necessary to handle the new factors.

Hardware Approach to Expanding

Adding a new component may require adding additional hardware. However, the addition of more hardware depends on the physical resources already available, plus what is needed for the new component.

If the number of subscribers exceeds 10 million, then the customer needs to Clone and Repartition sessionmgr Disks. See Manage Disks to Accommodate Increased Subscriber Load, on page 16.

High Availability Consequences

When adding more hardware, the design must take into consideration the high availability (HA) needs of the system. The HA design for a single-site system is N+1 at the hardware and application level. As a result, adding a new blade incrementally increases the HA capacity of the system.

For example, in a basic installation there are 2 Cisco Policy Server blades handling the traffic. The solution is designed so that if one of the blades fails, the other blade can handle the entire capacity of the system. When adding a third blade for capacity expansion, there are now 2 blades to handle the system load if one of the blades fails. This allows for a more linear scaling approach because each additional blade can be accountable for being able to use its full capacity.

Note When adding new blades to a cluster, the blades in the cluster must be co-located to achieve the proper throughput between other components.

Adding a New Blade

Step 1 Install ESX server to the blade.

Step 2 Open the CPS Deployment Template spreadsheet. This spreadsheet should have been created and maintained during the initial deployment.

Step 3 In the Additional Hosts sheet, add an entry for the new ESX server with IP, Host name and Alias.

Step 4 Save the CSV file and transfer it to the following directory on the Cluster Manager /var/qps/config/deploy/csv

Step 5 Run `/var/qps/install/current/scripts/import/import_deploy.sh` to convert the csv to json.

Component (VM Node) Approach to Expanding

The most common components to be expanded are on the Cisco Policy Servers. As your system begins to scale up, you will need to add more CPS nodes and more SessionMgrs. Expansion for other components can follow the same pattern as described here. The next sections discuss the configurations needed for those specific components to be active in the system.

Adding Additional Component

Step 1 Modify the CPS Deployment Template spreadsheet (this spreadsheet should have been created and maintained during the initial deployment).

Step 2 In the Hosts sheet, add the new VM node with the parameters. See the *CPS Installation Guide for VMware* for details about each column.

Step 3 Save the CSV file and transfer it to the following directory on the Cluster Manager: `/var/qps/config/deploy/csv`.

Step 4 Run `/var/qps/install/current/scripts/import/import_deploy.sh` to convert the csv to json.

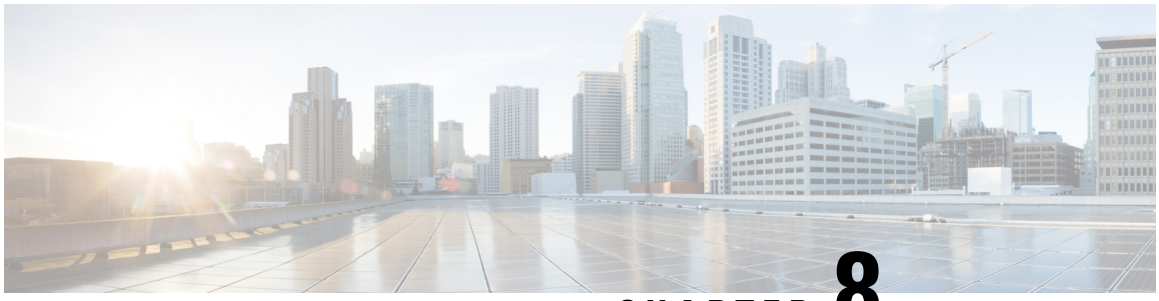
Step 5 Deploy the new VM using `/var/qps/install/current/scripts/deployer/deploy.sh` xxx, where xxx is the alias of the new VM to be deployed.

Refer to the *CPS Installation Guide for VMware* for more details about using `deploy.sh`.

Enable SSL

CPS uses encryption on all appropriate communication channels in HA deployments. No additional configuration is required.

Default SSL certificates are provided with CPS but we recommend that you replace these with your own SSL certificates. Refer to Replace SSL Certificates in the *CPS Installation Guide for VMware* for more information.



CHAPTER 8

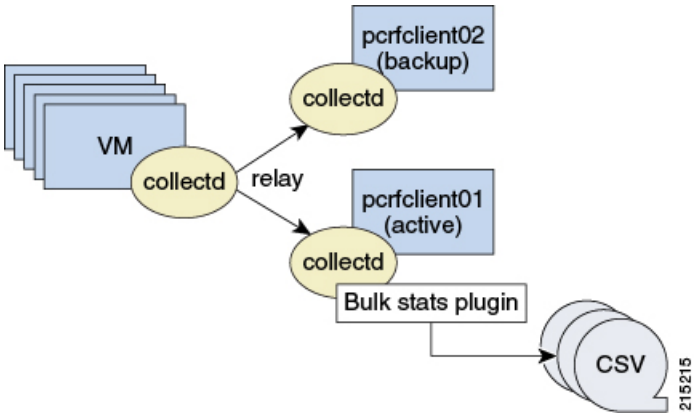
CPS Statistics

- Bulk Statistics Overview, on page 139
- CPS Statistics, on page 140
- Bulk Statistics Collection, on page 144
- CPS KPI Monitoring, on page 146
- Example CPS Statistics, on page 169

Bulk Statistics Overview

Bulk Statistics are the statistics that are gathered over a given time period and written to a set of files. These statistics can be used by external analytic processes and/or network management systems. The architecture of CPS bulk statistic collection is shown below.

Figure 56: CPS Bulk Statistic Collection Architecture



The collection utility collectd is used for collecting and storing statistics from each VM. Detailed collectd documentation can be found on http://collectd.org/.

Collectd within CPS is deployed with nodes relaying data using the collectd network plug-in (https://collectd.org/wiki/index.php/Plugin:Network) to the centralized collection nodes on the pcrfclient01 and pcrfclient02 virtual machines. The centralized collector writes the collected data to output CSV files.

Note pcrfclient01 and pcrfclient02 collect bulk statistics independently. As a result, it is normal to have slight differences between the two files. For example, pcrfclient01 generates a file at time t and pcrfclient02 generates a file at time t +/- the clock drift between the two machines.

As a best practice, always use the bulk statistics collected from pcrfclient01. pcrfclient02 can be used as a backup if pcrfclient01 fails.

If pcrfclient01 becomes unavailable, statistics is still gathered on pcrfclient02. Statistics data is not synchronized between pcrfclient01 and pcrfclient02, so a gap exists in the collected statistics while pcrfclient01 is down.

Note Statistics value in csv files is displayed in E notation format depending on value and data source type. For example, for Gauge type of data source, statistics value is converted to E notation if value is greater than 10^7.

Grafana

CPS Statistics

The list of statistics available in CPS is consolidated in an Excel spreadsheet. After CPS is installed, this spreadsheet can be found in the following location on the Cluster Manager VM:

```
/var/qps/install/current/scripts/documents/QPS_statistics.xlsx
```

Overview

The following diagram represents the various statistic gathering points for incoming and outgoing messages.

Figure 57: Various Statistic Gathering Points for Incoming and Outgoing Messages

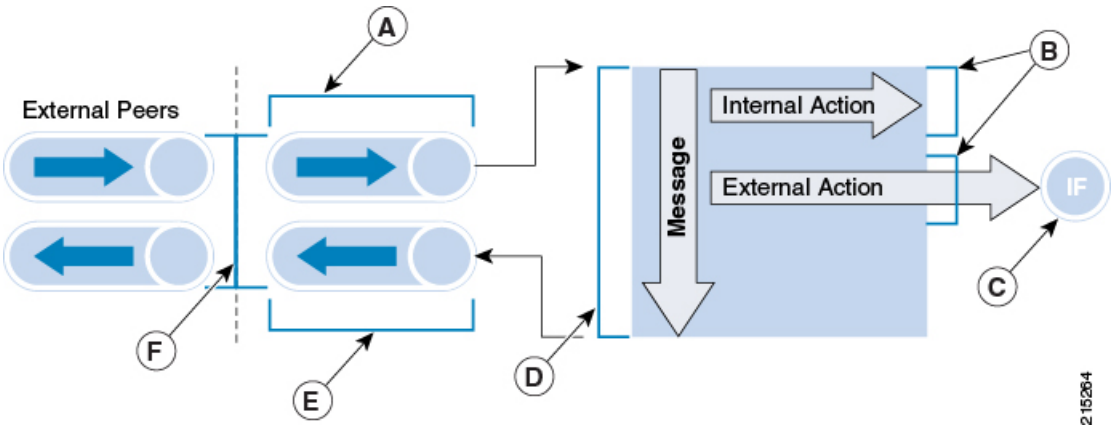


Table 14: Measurement Legend

Legend	Description
A	Inbound queue counts and times*
B	Policy action counts and times
C	Interface specific counts and times
D	Policy message counts and times
E	Outbound queue counts and times*
F	Round trip counts and times*
where, * – statistics only apply to Diameter messages	

A brief description of each statistic gathering points is given below:

- Upon receipt of a message on the Policy Director (lb) node, the message is registered as received and forwarded to a middle tier processing node.
- This middle tier processing node tracks the inbound message counts and time spent within the inbound processing queue. If a message is discarded due to SLA violation, then counters are incremented at this point. This occurs at point A within the diagram.
- Upon arrival within the policy engine all messages are counted and timers are started to measure the duration of processing.
- Any internal or external actions are tracked at this point and the round trip time is measured from the policy engine invocation of the action and success or failure of the action. This occurs at point B within the diagram.
- For external actions (for example, LDAP), interface specific statistics maybe captured. This occurs at point C in the diagram and is gathered from the Policy Director nodes.
- Upon completion of the message in the policy engine, the total elapsed time is measured and whether success or failure occurred in processing.

Note A message is considered a success even if the policy returns an error (such as 5002). These application errors are tracked at point D within the diagram.

- Outbound messages are tracked from the policy engine to the Policy Directors at point E within the diagram.
- Upon receipt of outbound messages, the Policy Directors tracks either end to end completion time for inbound requests OR starts a timer and counts outbound requests. This occurs at point F within the diagram.

CPS Statistic Types

This section describes various forms of statistics generated by CPS.

Diameter Statistics

In Diameter statistics, Monitoring Areas are defined on the basis of Queues maintained in it. Diameter statistics can also be defined based on whether the statistic is related to a counter or gauge or derived or absolute.

- Counter: Counter type represents a non-negative integer which monotonically increases until it reaches a maximum value of 2^32-1 (4294967295 decimal), when it resets and starts increasing again from zero.

  Counters have no defined “initial” value, and thus, a single value of a Counter has (in general) no information content. You must take a delta of multiple readings to understand anything.

- Gauge: Gauge type represents a non-negative integer, which can increase or decrease, but can never exceed a maximum value, nor fall below a minimum value. The maximum value cannot be greater than 2^32-1 (4294967295 decimal), and the minimum value cannot be smaller than 0.

- Derived: It is intended to store the derivative of the line going from the last to the current value of the data source. Such data sources are very common with events that can be counted. Internally, derive works exactly like COUNTER but without overflow checks. So if your counter does not reset at 32 or 64 bit you might want to use DERIVE and combine it with a MIN value of 0.

- Absolute: It is intended for counters which get reset upon reading. In effect, the type is very similar to GAUGE except that the value is an (unsigned) integer and is divided by the time since the last reading. This is used for fast counters which tend to overflow. So instead of reading them normally you reset them after every read to make sure you have a maximum time available before the next overflow. Another usage is for things you count like number of messages since the last update.

LDAP Statistics

CPS tracks LDAP statistics for general LDAP actions, LDAP query counters, LDAP connection counters, as well as message counters.

Categories:

- Action
- Messages

System Statistics

System statistics are defined based on six categories:

- CPU
- File System Usage
- Disk Performance
- Interface
- CPU Load
- Memory

Engine Statistics

Engine statistics are defined based on three categories:

- Session Count
- Session Operation
- Internal messages

MOG API Statistics

API statistics are defined based on five categories: Bearer Count, Tenant Onboarding Count, Subscriber Onboarding Count, Authentication Count and Callback Response Statistics.

Default and Dedicated Bearer Counters

Counter for the number of default and dedicated bearers related to API requests.

Default and Dedicated Bearer Statistics

Provides the statistics for default and dedicated bearers related to API requests.

Tenant Onboarding Counters

Counter for the number of tenant onboarding related to API requests.

Tenant Onboarding Statistics

Provides the statistics for tenant onboarding related to API requests.

Subscriber Onboarding Counters

Counter for the number of subscriber onboarding related to API requests.

Subscriber Onboarding Statistics

Provide the statistics for subscriber onboarding related to API requests.

Error Statistics Definitions

About error statistics, here are the definitions of each error suffix:

Table 15: Error Statistics Definitions

Error Statistics	Description
node1.messages.*.error	Failure processing a message
e2e*_qns_stat.error	Count of occurrence for given Diameter result code
pe-submit-error	Error submitting to policy engine
_bypass	Message not sent to policy engine due to successful response (2001)

Error Statistics	Description
_drop	Message dropped due to SLA violation
rate-limit	Message dropped due to rate limiting violation

Note The Diameter E2E statistics with the suffix “error” always have a value of 0 (zero) unless they have “_late” in the statistic name.

Bulk Statistics Collection

By default, CPS outputs a bulk statistics CSV file to the /var/broadhop/stats/ directory on the pcrfclient01 and pcrfclient02 VMs in five minute intervals.

The default naming standard is bulk-hostname-YYYY-MM-DD-HH-MI.csv

These CSV files include all statistics collected from all VMs during the 5 minute interval.

Note If a statistic is generated by the system multiple times within the 5 minute interval, only the last measured statistic is collected in the CSV file.

The following list is a sample of the file names created in the /var/broadhop/stats/ directory on the pcrfclient01 VM.

```
[root@pcrfclient01 stats]# pwd
/var/broadhop/stats
[root@pcrfclient01 stats]# ls
bulk-pcrfclient01-201510131350.csv
bulk-pcrfclient01-201510131355.csv
bulk-pcrfclient01-201510131400.csv
bulk-pcrfclient01-201510131405.csv
bulk-pcrfclient01-201510131410.csv
bulk-pcrfclient01-201510131415.csv
bulk-pcrfclient01-201510131420.csv
bulk-pcrfclient01-201510131425.csv
bulk-pcrfclient01-201510131430.csv
bulk-pcrfclient01-201510131435.csv
bulk-pcrfclient01-201510131440.csv
bulk-pcrfclient01-201510131445.csv
bulk-pcrfclient01-201510131450.csv
bulk-pcrfclient01-201510131455.csv
bulk-pcrfclient01-201510131500.csv
bulk-pcrfclient01-201510131505.csv
bulk-pcrfclient01-201510131510.csv
bulk-pcrfclient01-201510131515.csv
bulk-pcrfclient01-201510131520.csv
bulk-pcrfclient01-201510131525.csv
bulk-pcrfclient01-201510131530.csv
bulk-pcrfclient01-201510131535.csv
bulk-pcrfclient01-201510131540.csv
bulk-pcrfclient01-201510131545.csv
bulk-pcrfclient01-201510131550.csv
bulk-pcrfclient01-201510131555.csv
```

```
bulk-pcrfclient01-201510131600.csv
bulk-pcrfclient01-201510131605.csv
bulk-pcrfclient01-201510131610.csv
bulk-pcrfclient01-201510131615.csv
bulk-pcrfclient01-201510131620.csv
bulk-pcrfclient01-201510131625.csv
bulk-pcrfclient01-201510131630.csv
```

Retention of CSV Files

CPS retains each bulk statistic CSV file on the pcrfclient01/02 VM for 2 days, after which the file is automatically removed. If you need to preserve these CSV files, you must back up or move them to an alternate system.

Configuring Logback.xml

Configuration of the CPS application statistics is controlled in the `/etc/collectd.d/logback.xml` file.

Refer to http://logback.qos.ch/manual/appenders.html for more information about the configuration of the `logback.xml` file.

Collectd is configured in the following files:

- `/etc/collectd.conf`
- `/etc/collectd.d/jmxplugin.conf`
- `/etc/collectd.d/exec.conf`

Restarting the Collectd Service

After making any configuration changes to logback.xml, restart the collectd service:

```
monit restart collectd
```

Adding Realm Names to Diameter Statistics

By default, the Diameter statistics that are generated do not include the realm names. To include realms in the statistics collected, add the following line in the qns.conf file (comma separated auth-appl-id).

```
-Ddiameter.appid.realm.stats=Auth-Appl-Id-1,Auth-Appl-Id-2,… Auth-Appl-Id-n
```

where each Auth-Appl-Id refers to the specific protocol's Auth-Application-Id for which realms are needed in the statistics.

For example, to add Gx, Gy, Rx and Sy realms to the statistic names, use the following Auth-Appl-Ids:

```
-Ddiameter.appid.realm.stats=16777238,16777235,16777236,9
```

where

- Gx Auth-Application-ID = 16777238
- Rx Auth-Application-ID = 16777236
- Gy Auth-Application-ID = 4

- Sy Auth-Application-ID = 7

Note Adding a realm will increase the number of statistics generated/collected. Add realms only when necessary.

As an example, statistic names with and without the realms are shown below for reference for the following statistic:

e2e_<domain>_[realm_][alias_]<message id>

Counter name with Realm (with qns.conf file modification):

C,lb02,node2.messages.e2e_PHONE_sy-ac.cisco.com_AC_Syp_AAR_2001.qns_stat.success,528

C,lb02.node2.messages.e2e_PHONE_sy-bm.cisco.com_BM_Syp_AAR_2001.qns_stat.success,1221

Counter name without Realm (without qns.conf file modification):

C,lb01,node2.messages.e2e_PHONE_AC_Syp_AAR_2001.qns_stat.success,1495

C,lb01,node2.messages.e2e_PHONE_BM_Syp_AAR_2001.qns_stat.success,4

Each statistic field has a fixed maximum length of 63 characters. Based on the current syntax, the length of the realm should not exceed 16 characters, otherwise it will lead to truncation of the counter name.

CPS KPI Monitoring

This section provides a list of Key Performance Indicators (KPIs), useful for tracking the overall health of CPS.

The complete list of CPS statistics is available in a spreadsheet format in the following location on the Cluster Manager VM:

`/var/qps/install/current/scripts/documents/QPS_statistics.xlsx`

The KPIs highlighted in the following sections are also included on the **Stats Recommended to Monitor** tab in the `QPS_statistics.xlsx` spreadsheet.

System Health Monitoring KPIs

The following table lists the KPIs and thresholds to track the overall performance of the CPS deployment, including information about the underlying hardware.

Table 16: System Health Monitoring KPIs

Name/Description	Statistics/Formula	Warning Threshold	Major Threshold
CPU Utilization CPU is a critical system resource. When the demand increases and CPU utilization exceeds 80% utilization, the efficiency of the CPU is reduced. When CPU utilization exceeds 80%, the application processing time will increase, message response will increase, and drops and timeouts will be seen.	100 - cpu.<*cpuid*>.idle	> 60% utilization over 60 second period (assuming that idle is less than 40%)	> 80% utilization over 60 second period (assuming idle is less than 20%)
CPU Steal If multiple VMs on the same hypervisor and same hardware have concurrent CPU demands, the hypervisor will "steal" CPU from one VM to satisfy another VM CPU needs. If the CPU Steal statistic is non-zero, there is not enough CPU allocated for the VMs.	cpu.<*cpuid*>.steal	-	> 2% over 60 second period
CPU I/O Wait This monitors CPU I/O wait time. High CPU wait times may indicate CPUs waiting on disk access.	cpu.<*cpuid*>.wait	> 30 for more than 5 min	> 50 for more than 10 min
Memory utilization Memory is a system resource, which needs to be less than 80%. The swap threshold has been reduced for CPS, and swapping should occur when the system resources are exhausted and memory utilization hits 99%.	memory.free – memory.used	> 70% utilization over 60 second period	> 80% utilization over 60 second period
Disk Utilization Disk storage is a critical system resource, and when file system utilization exceeds 90% utilization the system can become less efficient. When the file system utilization hits 100%, then application can stop functioning.	df.<fs>.df_complex.free - df.<fs>.df_complex.used	> 80% utilization	> 90% utilization

Name/Description	Statistics/Formula	Warning Threshold	Major Threshold
Session Store utilization This KPI monitors the amount of database storage available. The data is evenly distributed across all shards, so any specific shard will have the same utilization rate as all shards.	var-data-sessions_1-free - var-data-sessions_1-used	> 70% utilization	> More than 80% utilization
In Queue These statistics monitors how long a message waits in the application queue, waiting to be serviced. The value should be 0 all the time. Non-zero values indicate the application is too slow, short of resources, or overwhelmed.	node1.messages.in_q*.avg	-	More than 1 ms over 60 seconds
Database lock This KPI monitors if database locking is excessive in a platform. Database locking is normal and expected, but the locks should be quick and a very low percentage. Values higher than 20% indicate problems with the database operation or the system resources associated with the database such as disk or memory.	*lock.percent	> 15% lock	> 20% lock
Diameter 3xxx errors Diameter Too Busy 3xxx message indicate that the PCRF is overwhelmed, or responding too slowly. This can be related to In Queue issues, system resources, database problems, network latency, or issues with SPR or other external nodes in the call flow.	messages.e2e_*_ 3xxx.success (and exclude the late statistics) as a percentage of *.node*.messages. e2e_*2001.success	> 0.5% of *.node*.messages. e2e_*2001.success Over 30 minute period	> 1% of *.node*.messages. e2e_*2001.success Over 30 minute period
Diameter 5xxx errors Session Not Found and other Diameter 5xxx errors indicate a critical problem with the ability to process the incoming diameter message. This can be related to exhausted PCRF system resources, invalid session id or bad message structure, length, or content, or even database corruption.	messages.e2e_*_ 5xxx.success (and exclude the late statistics) as a percentage of *.node*.messages. e2e_*2001.success	> 0.5% of *.node*.messages. e2e_*2001.success Over 5 minute period	> 1% of *.node*.messages. e2e_*2001.success Over 5 minute period

Name/Description	Statistics/Formula	Warning Threshold	Major Threshold
Diameter Message Response Time	-	> 100 ms for more than 30 minutes	> 300 ms for more than 15 minutes
Active Session Count	set_session_count_total. records	>80% of the lessor of the dimensioned or licensed capacity for more than 1 hour or = 0 for more than 5 minutes	>80% of the lessor of the dimensioned or licensed capacity for more than 10 minutes or = 0 for more than 10 minutes
Policy Execution Count (Internal TPS)	-	> 80% of the lessor of the dimensioned TPS capacity for more than 1 hour or = 0 for more than 5 minutes	> 80% of the lessor of the dimensioned TPS capacity for more than 10 minutes or = 0 for more than 10 minutes
Policy Errors	-	> 0	> 20 within 5 minutes
Dedicated Bearer Errors	node1.counters.<*domain*> _[realm_] Gx_bearer_setup_qci_<*qci*> _fail_<*failure-code*> .qns_count as a percentage of node1.counters. <*domain*> _[realm_] Gx_bearer_setup_qci_<*qci*>. qns_count	> .1	> .5

Name/Description	Statistics/Formula	Warning Threshold	Major Threshold
% of failed VoLTE calls due to resource allocation This KPI monitors failed VoLTE calls due to resource allocation errors on the PCEF. A spike in this measurement does not indicate a CPS issue, but may flag an issue in the mobile network that should be investigated.	-	> .1	> .5
% of Messages dropped due to SLA timeout Messages dropped due to SLA timeouts indicate that the PCRF is overwhelmed, or responding too slowly. This can be related to In Queue issues, system resources, database problems, network latency, or issues with SPR or other external nodes in the call flow.	node1.counters.[realm_]* _drop.qns_count as a percentage of *.node*.messages. e2e_*2001.success	> 0.5% of *.node*.messages. e2e_*2001.success	> 1% of *.node*.messages. e2e_*2001.success

Session Monitoring KPIs

The following KPIs enable you to monitor CPS session operation volumes, error counts and other useful statistics.

Note As each deployment is unique, no recommended ranges are provided. Cisco recommends monitoring these KPIs for a period of time (1-3 months) to establish a baseline. Deviations can then be monitored from the baseline values.

Table 17: Session Monitoring KPIs

Category	Name/Description	Statistics/Formula	Availability/ Node
Session Operation	Errored session creation count	node1.actions.CreateEntry. qns_stat.error	Policy Server (qns)
Session Operation	Successful session creation count	node1.actions.CreateEntry. qns_stat.success	Policy Server (qns)
Session Operation	Total milliseconds of successful session creations	node1.actions.CreateEntry. qns_stat.total_time_in_ms	Policy Server (qns)

Category	Name/Description	Statistics/Formula	Availability/ Node
Session Operation	Errored session deletion count	node1.actions.DeleteEntry. qns_stat.error	Policy Server (qns)
Session Operation	Successful session deletion count	node1.actions.DeleteEntry. qns_stat.success	Policy Server (qns)
Session Operation	Total milliseconds of successful session deletions	node1.actions.DeleteEntry. qns_stat.total_time_in_ms	Policy Server (qns)
Session Operation	Errored session retrieval count	node1.actions. GetSessionAction. qns_stat.error	Policy Server (qns)
Session Operation	Successful session retrieval count	node1.actions. GetSessionAction. qns_stat.success	Policy Server (qns)
Session Operation	Total milliseconds of successful session retrievals	node1.actions. GetSessionAction. qns_stat.total_ time_in_ms	Policy Server (qns)
Session Operation	Errored session update count	node1.actions.UpdateEntry. qns_stat.error	Policy Server (qns)
Session Operation	Successful session update count	node1.actions.UpdateEntry. qns_stat.success	Policy Server (qns)
Session Operation	Total milliseconds of successful session updates	node1.actions.UpdateEntry. qns_stat.total_ time_in_ms	Policy Server (qns)
Internal Messages	Errored timer messages	node1.messages. TimerExpired. qns_stat.error	Policy Server (qns)
Internal Messages	Successful timer messages	node1.messages. TimerExpired. qns_stat.success	Policy Server (qns)

Category	Name/Description	Statistics/Formula	Availability/ Node
Session	Gauge count of lock percentage	<*set_name*>. lock.percent	sessionmgr
Session	Gauge count of delete operations	<*set_name*>. op_delete.gauge	sessionmgr
Session	Gauge count of insert operations	<*set_name*>. op_insert.gauge	sessionmgr
Session	Gauge count of update operations	<*set_name*>. op_update.gauge	sessionmgr
Secondary Key Operations	Per ring count of failed lookup for primary key using the secondary key in cache ring	node1.counters.skcache_ring <*1|2*>_cache_miss. qns_count	Policy Server (qns)
Session Type Count	Count of session types (GX_TGPP/RX_TGPP/ SY_PRIME/SD_V11 … etc) in active session DB partition per admin set	<*setid*>.set_ <*set number of admin db*> _session_type_ <*session_type*>.records	sessionmgr
Session Count	Count of sessions in all active session DB partitions Threshold: > 80% of dimensioned or licensed capacity for more than 1 hour, or = 0 (zero) for more than 5 minutes	set_session_count_ total.records	Policy Server (qns)

Diameter Monitoring KPIs

The following CPS KPIs are useful for monitoring Diameter message traffic.

Note As each deployment is unique, no recommended ranges are provided. Cisco recommends monitoring these KPIs for a period of time (1-3 months) to establish a baseline. Deviations can then be monitored from the baseline values.

Table 18: Diameter Monitoring KPIs

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-I_2001.qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-I_2001.qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-I_3xxx.qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-I_4xxx.qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-I_5xxx.qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters.[realm_]Gx_CCR-I.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-U_2001.qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-U_2001.qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-U_3xxx.qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-U_4xxx.qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-U_5xxx.qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters.[realm_]Gx_CCR-U.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_CCR-U_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_CCR-U. qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-T_2001.qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-T_2001.qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-T_3xxx.qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-T_4xxx.qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Gx_CCR-T_5xxx.qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/A	Diameter Input Queue	node1.counters.[realm_]Gx_CCR-T.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Gx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Gx_RAR_timeout. qns_stat.success	Success timeout count for RAR message	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Gx/A	Diameter Input Queue	node1.counters. [realm_] Gx_RAA.qns_count	Count of all messages sent to the policy engine	Policy Server (qns)
Gx/A	Diameter Input Queue	node1.messages. in_q_Gx_RAA. qns_stat.error	Count of messages failed to be sent to the policy engine	Policy Server (qns)
Gx/A	Diameter Input Queue	node1.messages. in_q_Gx_RAA. qns_stat.success	Count of messages successful sent to the policy engine	Policy Server (qns)
Gx/E	Diameter Output Queue	node1.counters. [realm_] Gx_RAR.qns_count	Count of messages successful sent to the Policy Director (LB)	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_AAR_timeout. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_RAA.qns_count	Count of messages successful sent to the Policy Director (LB)	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_AAR_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Rx/E	Diameter Output Queue	node1.counters. [realm_] Rx_AAA_2001. qns_count	Count of AAA messages with result-code = 2001 sent successfully to the Policy Director (LB)	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_2001.qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_2001.qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_3xxx.qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_4xxx.qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_5xxx.qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages.e2e_<domain>_[realm_]Rx_ASR_retry.qns_count	Retry count for ASR message	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_ASA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_]Rx_ASA. qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_ASA_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_2001. qns_stat.total_ time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_RAR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_RAA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_RAA.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_RAA_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_2001. qns_stat.total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Rx/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Rx_STR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_STR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.counters. [realm_] Rx_STR_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.messages. in_q_Rx_STR. qns_stat.success	Count of messages successful sent to the policy engine	Policy Server (qns)
Rx/A	Diameter Input Queue	node1.messages. in_q_Rx_STR. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Rx/D	Engine Message	node1.messages. diameter_Rx_STR. qns_stat.success	Success message count	Policy Server (qns)
Rx/D	Engine Message	node1.messages. diameter_Rx_STR. qns_stat. total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)
Rx/E	Diameter Input Queue	node1.counters. [realm_] Rx_STA_2001. qns_count	Count of STA messages with result-code = 2001 sent successfully to the Policy Director (LB)	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SLR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SLR_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SLR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SLR_drop.qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_SLA. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_SLA. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_SLA. qns_stat.success	Success message count	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_SLA. qns_stat. total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Sy/B	Diameter Action	node1.actions. send.diameter_ Sy_SLR.qns_stat.success	Success actions count	Policy Server (qns)
Sy/B	Diameter Action	node1.actions. send.diameter_ Sy_SLR.qns_stat. total_time_in_ms	Total milliseconds of successful actions	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_SNR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SNR.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_SNR_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_ Sy_SNR. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_SNR. qns_stat. total_time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_2001. qns_stat.success	Success message count for return code 2001	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_2001. qns_stat. total_time_in_ms	Total milliseconds of successful messages with return code matching 2001	Policy Director

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_3xxx. qns_stat.success	Success count of messages with return code matching 3XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_4xxx. qns_stat.success	Success count of messages with return code matching 4XXX	Policy Director
Sy/F	Diameter Round Trip	node[x].messages. e2e_<domain>_ [realm_] Sy_STR_5xxx. qns_stat.success	Success count of messages with return code matching 5XXX	Policy Director
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA_bypass. qns_count	Count of message that do not require processing by the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA.qns_count	Count of messages successful sent to the policy engine	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.counters. [realm_] Sy_STA_drop. qns_count	Count of messages dropped due to exceeding SLA	Policy Server (qns)
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_STA. qns_stat.success	Count of messages successfully sent to the policy engine	Policy Server (qns)

AppId/ Monitoring Area	Category	Statistic	Description	Availability/ Node
Sy/A	Diameter Input Queue	node1.messages. in_q_Sy_STA. qns_stat.total_ time_in_ms	Total milliseconds of messages successfully sent to the policy engine	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_STA. qns_stat.success	Success message count	Policy Server (qns)
Sy/D	Engine Message	node1.messages. diameter_Sy_STA. qns_stat.total_time_in_ms	Total milliseconds of successful messages	Policy Server (qns)
Sy/B	Diameter Action	node1.actions.send. diameter_Sy_STR. qns_stat.success	Success actions count	Policy Server (qns)
Sy/B	Diameter Action	node1.actions.send. diameter_ Sy_STR.qns_stat. total_time_in_ms	Total milliseconds of successful actions	Policy Server (qns)
Sy/E	Diameter Output Queue	node1.counters. [realm_] Sy_STR.qns_count	Count of messages successfully sent to the Policy Director (LB)	Policy Server (qns)

Example CPS Statistics

Sample CSV Files

The following list is a sample of the file names created in the /var/broadhop/stats directory on the pcrfclient01 VM.

```
[root@pcrfclient01 stats]# pwd
    /var/broadhop/stats
    [root@pcrfclient01 stats]# ls
    bulk-pcrfclient01-201510131350.csv
    bulk-pcrfclient01-201510131355.csv
    bulk-pcrfclient01-201510131400.csv
    bulk-pcrfclient01-201510131405.csv
    bulk-pcrfclient01-201510131410.csv
```

```
bulk-pcrfclient01-201510131415.csv
bulk-pcrfclient01-201510131420.csv
bulk-pcrfclient01-201510131425.csv
bulk-pcrfclient01-201510131430.csv
bulk-pcrfclient01-201510131435.csv
bulk-pcrfclient01-201510131440.csv
bulk-pcrfclient01-201510131445.csv
bulk-pcrfclient01-201510131450.csv
bulk-pcrfclient01-201510131455.csv
bulk-pcrfclient01-201510131500.csv
bulk-pcrfclient01-201510131505.csv
bulk-pcrfclient01-201510131510.csv
bulk-pcrfclient01-201510131515.csv
bulk-pcrfclient01-201510131520.csv
bulk-pcrfclient01-201510131525.csv
bulk-pcrfclient01-201510131530.csv
bulk-pcrfclient01-201510131535.csv
bulk-pcrfclient01-201510131540.csv
bulk-pcrfclient01-201510131545.csv
bulk-pcrfclient01-201510131550.csv
bulk-pcrfclient01-201510131555.csv
bulk-pcrfclient01-201510131600.csv
bulk-pcrfclient01-201510131605.csv
bulk-pcrfclient01-201510131610.csv
bulk-pcrfclient01-201510131615.csv
bulk-pcrfclient01-201510131620.csv
bulk-pcrfclient01-201510131625.csv
bulk-pcrfclient01-201510131630.csv
```

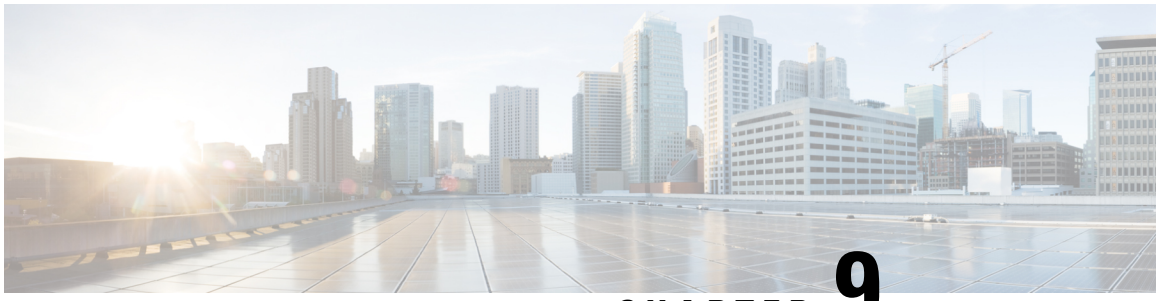
Sample Output

C,<*VM_name*>,node1.actions.send.diameter_Gx_CCA-I.qns_stat.success,19

where, the <*VM_Name*> indicates which VM the statistics has been collected on.

A sample bulk statistics .csv file is shown below:

```
C,qns01,node1.actions.SaveSubscriberActionImpl.qns_stat.error,0
C,qns01,node1.actions.SaveSubscriberActionImpl.qns_stat.success,6
C,qns01,node1.actions.send.diameter_Gx_CCA-I.qns_stat.error,0
C,qns01,node1.actions.send.diameter_Gx_CCA-I.qns_stat.success,19
C,qns01,node1.actions.send.diameter_Gx_CCA-T.qns_stat.error,0
C,qns01,node1.actions.send.diameter_Gx_CCA-T.qns_stat.success,9
D,qns01,node1.messages.in_q_Gx_CCR-I.qns_stat.total_time_in_ms,14
D,qns01,node1.messages.in_q_Gx_CCR-T.qns_stat.total_time_in_ms,2
D,qns01,node1.messages.in_q_Gx_CCR-U.qns_stat.total_time_in_ms,1
D,qns01,node1.messages.in_q_Gx_RAA.qns_stat.total_time_in_ms,0
D,qns01,node1.messages.in_q_Sh_SNA.qns_stat.total_time_in_ms,2
D,qns01,node1.messages.in_q_Sh_UDA.qns_stat.total_time_in_ms,0
D,qns01,node1.messages.TimerExpired.qns_stat.total_time_in_ms,7244
D,qns01,node1.spr.createSubscriber.qns_stat.total_time_in_ms,29
D,qns01,node1.spr.deleteSubscriber.qns_stat.total_time_in_ms,40
D,qns01,node1.spr.getSubscriber.qns_stat.total_time_in_ms,44
D,qns01,node1.spr.updateSubscriber.qns_stat.total_time_in_ms,21
G,lb02,node1.ldap.SITELDAP.qns_ldap_connection.MaximumAvailableConnections,10.0
G,lb02,node1.ldap.SITELDAP.qns_ldap_connection.NumAvailableConnections,0.0
G,lb02,node1.thread.gauge.daemon_thread_count,80.0
G,lb02,node1.thread.gauge.live_thread_count,184.0
```



CHAPTER 9

Working with CPS Utilities

- Policy Tracing and Execution Analyzer, on page 171
- Network Cutter Utility, on page 175
- Policy Builder Configuration Reporter, on page 175
- CRD Generator Conversion Tool, on page 176
- Policy Builder Configuration Converter Conversion Tool, on page 179
- Modifying Audit Rule File, on page 180

Policy Tracing and Execution Analyzer

Cisco Policy Server comes with a set of utilities to actively monitor and trace policy execution. These utilities interact with the core policy server and the mongo database to trigger and store traces for specific conditions.

Architecture

The policy tracing and execution analyzer is 3-tier architecture:

- Tier 1 — command line utilities to manage the policy trace generation and extract policy traces.
- Tier 2 — policy server creation of policy traces using triggers defined in Tier 1.
- Tier 3 — storage of the policy traces in a MongoDB.

Administering Policy Traces

All commands are located on the Control Center virtual machine within `/var/qps/bin/control` directory. There are two main scripts which can be used for tracing: `trace_ids.sh` and `trace.sh`.

- The `trace_ids.sh` script maintains all rules for activating and deactivating traces within the system.
- The `trace.sh` script allows for the real time or historical retrieval of traces.

Before running `trace_ids.sh` and `trace.sh`, confirm which database you are using for traces. For more information, refer to Policy Trace Database, on page 174. If no database has been configured, then by default the scripts connects to primary database member of SPR-SET1.

Managing Trace Rules using trace_ids.sh

Running `trace_ids.sh` with *-h* arguments produces a help text describing the capabilities of the script.

```
/var/qps/bin/control/trace_ids.sh -h
```

Usage:

```
/var/qps/bin/control/trace_ids.sh -i <specific id> -d sessionmgr01:27719/policy_trace
   /var/qps/bin/control/trace_ids.sh -r <specific id> -d sessionmgr01:27719/policy_trace
    /var/qps/bin/control/trace_ids.sh -x -d sessionmgr01:27719/policy_trace
    /var/qps/bin/control/trace_ids.sh -l -d sessionmgr01:27719/policy_trace
```

Note By default , if *-d* option is not provided then the script connects to primary database member of SPR-SET1. If you are not using the SPR database, you need to find out the which database you are using. To find out which database you are using, refer to Policy Trace Database, on page 174. Make sure to update the commands mentioned in Step 1, on page 172 to Step 4, on page 172 accordingly.

This script starts a selective trace and outputs it to standard out.

Step 1 Specific audit ID tracing:

```
/var/qps/bin/control/trace_ids.sh -i <specific id>
```

Step 2 Remove trace for specific audit ID:

```
/var/qps/bin/control/trace_ids.sh -r <specific id>
```

Step 3 Remove trace for all IDs:

```
/var/qps/bin/control/trace_ids.sh -x
```

Step 4 List all IDs under trace:

```
/var/qps/bin/control/trace_ids.sh -l
```

Adding a specific audit ID for tracing requires running the command with the *-i* argument and passing in a specific ID. The policy server matches the incoming session with the ID provided and compares this against the following network session attributes:

- Credential ID
- Framed IPv6 Prefix
- IMSI
- MAC Address
- MSISDN
- User ID

If an exact match is found then the transaction are traced. Spaces and special characters are not supported in the audit ids.

- Removing a specific audit id from active tracing requires specifying the *-r* argument with id to remove.

- Removing all ids requires sending in the *-x* argument and this will remove all ids from the database.
- Listing all ids requires sending in the *-l* argument.

Usage:

Usage with SPR-SET as database:

```
#./trace_ids.sh -l
MongoDB shell version: 2.6.3
connecting to: sessionmgr01:27720/policy_trace
112345
MongoDB shell version: 2.6.3
connecting to: sessionmgr01:27720/policy_trace
null
```

Usage with *-d* option:

```
#./trace_ids.sh -l -d sessionmgr01:27717/policy_trace
MongoDB shell version: 2.6.3
connecting to: sessionmgr01:27717/policy_trace
874838
MongoDB shell version: 2.6.3
connecting to: sessionmgr01:27717/policy_trace
null
```

Situations where traces are generated automatically

The following criteria cause the system to generate a trace regardless of whether the id is present in the trace database or not:

- If there is an AVP with the code: audit_id, audit-id, auditid. In this case, the traces are stored in the database with the value of the AVP.
- If there is a subscriber attribute (USuM AVP) with a code of audit-policy and a value of "true". In this case, the traces are stored using the credentials stored for the subscriber.
- If an error is triggered internally.

Note An error is defined as an internal processing error (e.g. database failure or other failure) and is not a failure message code.

Managing Trace Results using trace.sh

Running `trace.sh` with *-h* arguments produce a help text describing the capabilities of the script:

```
/var/qps/bin/control/trace.sh -h
```

Usage:

```
/var/qps/bin/control/trace.sh -i <specific id> -d sessionmgr01:27719/policy_trace
    /var/qps/bin/control/trace.sh -x <specific id> -d sessionmgr01:27719/policy_trace
    /var/qps/bin/control/trace.sh -a -d sessionmgr01:27719/policy_trace
    /var/qps/bin/control/trace.sh -e -d sessionmgr01:27719/policy_trace
```

Note By default , if *-d* option is not provided then the script connects to primary database member of SPR-SET1. If you are not using the SPR database, you need to find out the which database you are using. To find out which database you are using, refer to Policy Trace Database, on page 174. Make sure to update the commands mentioned in Step 1, on page 174 to Step 4, on page 174 accordingly.

This script starts a selective trace and outputs it to standard out.

Step 1 Specific audit ID tracing:

```
/var/qps/bin/control/trace.sh -i <specific id>
```

Specifying the *-i* argument for a specific ID causes a real time policy trace to be generated while the script is running. Users can redirect this to a specific output file using standard Linux commands.

Step 2 Dump all traces for specific audit ID:

```
/var/qps/bin/control/trace.sh -x <specific id>
```

Specifying the *-x* argument with a specific ID, dumps all historical traces for a given ID. Users can redirect this to a specific output file using standard Linux commands.

Step 3 Trace all:

```
/var/qps/bin/control/trace.sh -a
```

Specifying the *-a* argument causes all traces to output in real time policy trace while the script is running. Users can redirect this to a specific output file using standard Linux commands.

Step 4 Trace all errors:

```
/var/qps/bin/control/trace.sh -e
```

Specifying the *-e* argument causes all traces triggered by an error to output in real time policy trace while the script is running. Users can redirect this to a specific output file using standard Linux commands.

Policy Trace Database

The default location of the policy trace database is the administrative database and can be optionally specified in the trace database fields. These fields are defined at the cluster level in the system configurations.

Note Make sure to run all trace utility scripts from `/var/qps/bin/control` directory only.

Configure Traces Database in Policy Builder

Step 1 Log in to the Policy Builder.

Step 2 From left pane, open up the *name of your system* and select the required cluster.

Step 3 From right pane, select the check box for **Trace Database**.

The following table provides the parameter descriptions under **Trace Database** check box:

Table 19: Trace Database Parameters

Parameter	Description
Primary Database IP Address	The IP address of the sessionmgr node that holds trace information which allows for debugging of specific sessions and subscribers based on unique primary keys.
Secondary Database IP Address	The IP address of the database that provides fail over support for the primary database. This is the mirror of the database specified in the Primary IP Address field. Use this only for replication or replica pairs architecture. This field is present but deprecated to maintain downward compatibility.
Database Port	Port number of the database for Session data. Default value is 27717.

Network Cutter Utility

CPS supports a new network cutter utility, which keeps monitoring Policy Server (QNS) VMs failures. When any of the Policy Server VMs are down, utility cuts those unnecessary connections to avoid sending traffic to Policy Server VMs that are down, and this also results in avoiding timeouts.

This utility is started by `monit` on Policy Director (lb) VMs and keeps monitoring policy server VMs failures.

Utility stores log on `/var/log/broadhop/network-cutter.log` file.

You can verify the status of network cutter utility on lb01/02 VMs using `monit summary` and `network-cutter` status command:

```
monit summary | grep cutter
Process 'cutter' Running
```

```
service network-cutter status
network-cutter (pid 3735) is running
```

You can verify if network cutter utility has been started using `ps -ef | grep cutter` command:

```
ps -ef | grep cutter
root 6496 1 0 Feb18 ? 00:16:22 /usr/java/default/bin/java -jar
/var/broadhop/images/network-cutter.jar
```

Policy Builder Configuration Reporter

The Configuration-Reporter utility processes CPS Policy Builder configuration and report any missing cross-reference files and stale files. An option has also been provided to remove the stale files and missing cross-references in the XMI files from the configuration data in the utility.

Important This utility can be used before or after installation to check if customers have all the configuration files needed.

This reporting utility address the following concerns:

- Reports if there are any missing PB configuration files (.xmi files) and a summary of what those files are.
- Reports if there are any stale files and a summary of the same.

  Stale files are Service Option files whose corresponding Use Case Template files are missing.
- It also shows the missing configuration files on a per-file basis, showing the files that are referencing the missing files.
- Additionally, the customer can see all the different configuration objects and their quantity to see the variety of configurations they are using.
- Using -r option, utility creates a new archive file with cleaned XMI files (removes the stale files and missing cross-references from XMI files from the original configuration data).

To run the utility, perform the following steps:

1. Mount ISO on Cluster Manager if you unmounted the ISO after completing the CPS installation or upgrade.
2. Extract the release train into the temp directory:

   ```
   cd /tmp
   tar -zxvf /mnt/iso/app/install/release-train-xxx.tar.gz
   ```

   where, `release-train-xxx.tar.gz` is the release train version.
3. Go into Configuration-Reporter directory which is present inside utility directory of extracted utility.

   ```
   cd release-train-xxx/Utility/Configuration-Reporter
   ```
4. Execute jar using the following command:

   ```
   java -jar configuration-reporter.jar <pb-configuration-xmi-files-in-archive-form> [-r]
   ```

   where,

   - *<pb-configuration-xmi-files-in-archive-form>* is the name of the configuration file.
   - `[-r]` is an optional parameter and if specified will remove all the references of missing files from XMI files and stale files in the archive file and outputs the corrected archive as `filename_cleaned.zip|cps` (output file will have same extension as input file) on the same path where command runs.

CRD Generator Conversion Tool

CPS provides a CRD conversion tool which converts existing Balance and Quota templates PB configuration data to CRD Data. You can provide XMI files to the tool in the following ways:

- Use Import/Export tool to export CPS configuration as an archive file (.cps extension archive) and provide the same to the tool.

- Archive set of XMI files to .zip extension archive file.
- Provide directory path where XMI files are present as an input to the tool.

Important The conversion tool is used to convert all Balance and Quota template configuration data to CRD data. This helps to reduce the number of XMI files in the system and improves the performance.

Prerequisites:

The feature `com.broadhop.balance.crdbalance.feature` must be enabled so that CRD tables for Balance and Quota Template details are displayed in Policy Builder (as readonly) and Control Center. These CRD tables need to be present for importing the Balance and Quota CRD data which will be converted using the tool and Balance and Quota Templates XMIs present in Policy Builder.

To enable `com.broadhop.balance.crdbalance.feature`, add the feature in `/var/qps/current_config/etc/broadhop/pb/features` and `/var/qps/current_config/etc/broadhop/pcrf/features` files. For more information, refer to *Customize Features in the Deployment* section in *CPS Installation Guide for VMware*.

To run the utility, perform the following steps:

1. Mount ISO on Cluster Manager.
2. Extract release train into temp directory:

   ```
   tar -zxvf /mnt/iso/app/install/xxx.tar.gz /tmp/
   ```

3. Go to `CRD_generator_Utility` directory which is inside the utility directory of the extracted release train:

   ```
   cd /tmp/release-train-xxx/Utility/CRD_generator_Utility
   ```

4. Execute jar using the following command:

   ```
   java -jar com.broadhop.customreferencedata.generator-<svn-revision-number>-full.jar [-a
    <archive-file> | -d <directory>]
   ```

   The following table describes the various command line options:

Command Line Options	Description
-a	Option for passing zip archive file which contains XMI files.
-d	Option for passing directory path where XMI files are present.
-e	Generates .exportCrdInfo file with specified exportCRDversion. Valid Values 1 and 2 are described as follows: 1. 1 : Data-type validations will happen only during import of generated archive into CPS. 2. 2 : All validations will happen during import of generated archive into CPS and is the default value.

Command Line Options	Description
-h	Prints help.
-o	Object type for which CRD conversion needs to be performed. Default value is AccountBalanceTemplate.
-r	Removes duplicate CRD data. Valid Values 0 and 1 are described as follows: 1. 0 : De-duplication is disabled by default which means duplicate data will be part of generated CRD data files. 2. 1 : Keeps the first record is retained and skips the rest. De-duplication is enabled which means duplicate data will be not be part of generated CRD data.
-v	Validates CRD data against the schema (required field constraint validation). Default value is true which means validation of schema is enabled and CRD data record with missing required field value will not be part of generated CRD data files.
-xls	Generates XLS format files for CRD data. Default option is CSV format CRD data files.

The tool generates a ".crd" extension archive file containing ".exportCrdInfo" file and CRD tables data in CSV/XLS format which can be used by Import/Import All CRD functionality in CPS to import the CRD data into the system.

5. To view or edit the csv files, perform the following optional steps:
   1. View-only using Excel : In Excel, you can only view the csv files present in generated ".crd" archive where non-ASCII characters are present in it. In order to view non-ASCII characters which might be present in CRD Data, perform the following steps:
      1. Open a blank Excel file.
      2. Go to Menu option **Data** > **From Text File** and import the CRD table CSV file in which non-ASCII is present.
      3. A "Text Import Wizard" is displayed. Perform the following steps:
         1. Select Unicode (UTF-8) in File origin drop-down.
         2. Check **Comma** as delimiters.
         3. Do not perform any changes and click **Finish**.
         4. Click **Ok**.
      4. All non-ASCII characters are displayed correctly.

Note It is not recommended to edit generated CRD Table csv files containing non-ASCII characters in excel view.

2. View and edit using other editors (vi editor): You can view and edit csv files present in ".crd" archive file using editors such as vi editor even if the CRD data contains non-ASCII characters.

6. The generated ".crd" extension archive file needs to be imported as CRD Data into CPS which can be performed using the following options:
   - Use "_import" CRD API to import CRD data in CSV format.
   - Use "Import All" option in Control Center to import CRD data in CSV format.
   - Use "Import" option in Control Center to import CRD data in XLS format. This option enables you to import single XLS CRD data at a time.

Policy Builder Configuration Converter Conversion Tool

CPS provides a conversion tool to convert the balance references in the existing service configuration to CRD data string value to adopt the CRD table driven configuration solution. The tool can perform the following:

- Convert Account Balance template references present in existing Customer's PB Service configuration to CRD Data "Dynamic Reference Data Key" string value.
- An "-r" option is provided to clean up the following converted referenced data:
  - References to Account Balance template in Service Options, Use Case Templates and Use Case Options is removed in the output archive file configuration data.
  - Account Balance template and all Quota templates present in the original PB configuration data will not be part of output archive file.

Important This conversion tool is used to convert balance references in existing configuration data and clean Balance and Quota templates as part of result archive file. This helps in reducing the number of XMI files in configuration data.

To run the utility, perform the following steps:

1. Mount ISO on Cluster Manager.
2. Extract release train into temp directory:

   ```
   tar -zxvf /mnt/iso/app/install/xxx.tar.gz /tmp/
   ```

3. Go to `PB-Configuration-Converter_Utility` directory which is inside the utility directory of the extracted release train:

   ```
   cd /tmp/release-train-xxx/Utility/PB-Configuration-Converter_Utility
   ```

4. Execute jar using the following command:

```
java –jar pb-configuration-converter-<svn-revision-number>-full.jar [-a <archive-file>
| -d <directory> ] [-r]
```

1. You have the option to provide the XMI files input as an archive file or directory path in which all Policy Builder created XMI files are present. Select any one of the following mandatory options to run the command:
   - -a : Option for passing Archive file (.zip or .cps extension archive file).
   - -d : Option for passing directory path containing XMI files to process.
2. You can use "-r" option to perform cleanup operation for reducing the XMI files as follows:
   - Removes Account Balance template references from Service Option XMIs once the update of “Dynamic Reference Data Key” is performed.
   - Removes Account Balance template references from Use Case Template and Use Case Option XMI files.
   - Removes Account Balance template , One Time Quota template, Recurring Quota template and Rollover Quota template XMI files from PB configuration data in resulting archive file.

The tool generates an archive file named as “<input-file-name>_updated.<input-file-extension>” if it is an archive file input or "<input-directory-name>_updated.zip" if it is a directory file input.. It contains all the XMI files in the input file along with updated Service Option XMI files with a new field “dynamicRefDataKey” if there are references to Account Balance template object type.

Note The output archive file might not contain “.exportInfo” and “.exportRepositoryInfo” files as the tool only works on conversion of Service configuration balance reference data present in user input and copies all other input files in the output archive.

Modifying Audit Rule File

You need to modify the `audit.rules` file as described in the following section.

Step 1 Add new rules to `etc/audit/rules.d/audit.rules` file.

Sample Audit Rule File

```
cat audit.rules:
-D
-b 8192
-w /etc/group -p wa -k identity
-w /etc/passwd -p wa -k identity
-w /etc/shadow -p wa -k identity
```

Step 2 Restart the auditd service by executing the following command:

```
service auditd restart
```

After restart, the audit files are loaded in the `audit.rules` file `/etc/audit` location.

Step 3 Execute the following command to view all the implemented audit rules:

```
auditctl -l
```

Step 4 Execute the following command to verify if the audit rule file is correct and error-free:

```
auditctl -R /etc/audit/audit.rules
```

If the audit.rule file has any error, it is highlighted in the output.

Step 5 Monitoring of logging for critical file is done in the following location on cluster manager:

```
/var/log/audit/ audit.log
```

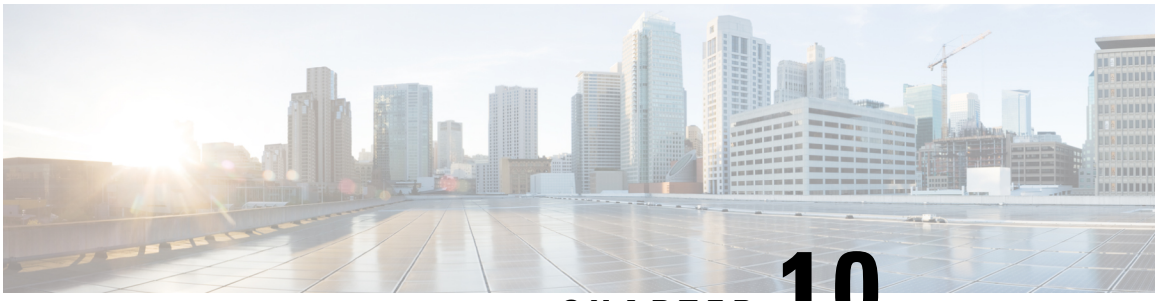
Step 6 Execute the following command to validate particular audit logs:

```
ausearch -i -k <key to moniter>
e.g. ausearch -i -k "identity"
---
time->Fri Aug 24 11:29:19 2018
type=PROCTITLE
msg=audit(1535110159.513:18131):proctitle=7573657264656C00736F6D61
type=PATH msg=audit(1535110159.513:18131): item=0 name="/etc/passwd"
inode=2886645 dev=08:02 mode=0100644 ouid=0 ogid=0 rdev=00:00 objtype=NORMAL
type=CWD msg=audit(1535110159.513:18131):  cwd="/root"
type=SYSCALL msg=audit(1535110159.513:18131): arch=c000003e syscall=2 success=yes
exit=5 a0=5595b6cd69e0 a1=20902 a2=0 a3=8 items=1 ppid=25552 pid=26946 auid=0 uid=0
gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts1 ses=984 comm="userdel"
exe="/usr/sbin/userdel" key="identity"
time->Fri Aug 24 11:45:23 2018
----
type=CONFIG_CHANGE msg=audit(1535111123.222:18175): auid=4294967295 ses=4294967295
op=remove_rule key="identity" list=4 res=1
```

Note You can add the following rules to the audit file to montior each file:

```
-w /var/run/utmp -p wa -k session

-w /var/log/wtmp -p wa -k session

-w /var/log/btmp -p wa -k session

-w /etc/login.defs -p wa -k password

-w /etc/group -p wa -k identity

-w /etc/passwd -p wa -k identity

-w /etc/shadow -p wa -k identity

-w /etc/sudoers -p wa -k identity

-w /etc/fstab -p wa -k quota_rate_limit

-w /etc/iptables -p wa -k network
```

`Identity`, `quota_rate_limit`, and `network` are keywords used to search the respective log.



CHAPTER 10

CPS Commands

Note If you want to save the output of any command on any of the CPS VMs, then the output should be redirected to a file.

- about.sh, on page 184
- adduser.sh, on page 184
- auditrpms.sh, on page 185
- build_all.sh, on page 185
- build_etc.sh, on page 187
- build_set.sh, on page 187
- capture_env.sh, on page 188
- change_passwd.sh, on page 189
- cleanup_license.sh, on page 189
- component_alarm_reports.py, on page 189
- copytoall.sh, on page 190
- diagnostics.sh, on page 191
- dump_utility.py, on page 194
- list_installed_features.sh, on page 198
- reinit.sh, on page 200
- restartall.sh, on page 200
- restartqns.sh, on page 201
- runonall.sh, on page 201
- service, on page 202
- session_cache_ops.sh, on page 202
- set_priority.sh, on page 206
- startall.sh, on page 207
- startqns.sh, on page 208
- statusall.sh, on page 208
- stopall.sh, on page 210
- stopqns.sh, on page 211
- summaryall.sh, on page 211
- sync_times.sh, on page 224
- syncconfig.sh, on page 225

- terminatesessions, on page 225
- top_qps.sh, on page 228
- vm-init.sh, on page 230

about.sh

This command displays:

- Core version
- Patch installed
- ISO version
- Feature version
- URLs to the various interfaces
- APIs for the deployment

Syntax

```
/var/qps/bin/diag/about.sh [-h]
```

Executable on VMs

- Cluster Manager
- pcrfclient01/02

adduser.sh

This utility adds a new user to the specified nodes that are part of the CPS deployment. These accounts will be provisioned without shell access and, as such, they're only useful for authenticating against the various web-based GUIs used to administrate CPS.

The hosts that get provisioned with these new accounts can be selected using the 'node-regex' option. The default regular expression used by the script is:

```
node-regex ::= ^(pcrfclient|qns|lb[0-9]+|sessionmgr)
```

Syntax

```
/var/qps/bin/support/adduser.sh [-h] [node-regex]
```

When prompted for the user's group, set 'qns-svn' for read-write permissions or 'qns-ro' for read-only permissions.

To add a user with 'read/write' access to Control Center, their group should be 'qns'.

- To check if a user already exists, login in as root and enter 'su <username>'.
- To check a user's 'groups', enter 'groups <username>'.

Executable on VMs

All

Example

```
[root@host /]# /var/qps/bin/support/adduser.sh
Enter username: username
Enter group for the user: groupname
Enter password: password
Re-enter password: password
The above example adds username to all the VMs in the cluster.
```

auditrpms.sh

This script runs in background on all VMs except Cluster Manager. This script/daemon should be always running and is monitored via monit. No intervention from end user is required. Corresponding logs are generated at individual nodes in `/var/log/broadhop/audit/audit_rpms.log`.

Note All successful attempts i.e. installation or removal are tracked in this file. In case package is upgraded there would be two entries seen in log file, one for removal of old package and one for installation of new package.

Executable on VMs

On all VMs except Cluster Manager

Example

```
[root@lb01 ~]# monsum | grep auditrpms
Process 'auditrpms.sh' Running
```

build_all.sh

This command is executed from Cluster Manager to rebuild CPS package.

Syntax

- `/var/qps/install/current/scripts/build_all.sh`
- `/var/qps/install/current/scripts/build/build_all.sh`

Executable on VMs

Cluster Manager

Example

```
[root@host /]# /var/qps/install/current/scripts/build_all.sh
Building /etc/broadhop...
Copying to /var/qps/images/etc.tar.gz...
```

```
Creating MD5 Checksum...
Copying /etc/puppet to /var/qps/images/puppet.tar.gz...
Creating MD5 Checksum...
Copying Policy Builder configuration (/var/qps/current_config/pb_config) to
/var/qps/images/svn.tar.gz...
Creating MD5 Checksum...
Updating tar from: /var/qps/env_config/ to /var/www/html/images/
Creating MD5 Checksum...
Building /var/qps/bin...
Copying /var/qps/bin to /var/qps/images/scripts_bin.tar.gz...
Creating MD5 Checksum...
Building images...
Building image: /var/qps/images/controlcenter.tar.gz
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.controlcenter.feature.feature.group
         com.broadhop.faultmanagement.service.feature.feature.group
         com.broadhop.infrastructure.feature.feature.group
         com.broadhop.server.runtime.product
         com.broadhop.snmp.feature.feature.group
Creating MD5 Checksum... /var/qps/images/controlcenter.tar.gz.md5chksum
Building image: /var/qps/images/diameter_endpoint.tar.gz
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.diameter2.service.feature.feature.group
         com.broadhop.server.runtime.product
         com.broadhop.snmp.feature.feature.group
Creating MD5 Checksum... /var/qps/images/diameter_endpoint.tar.gz.md5chksum
Building image: /var/qps/images/iomanager01.tar.gz
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.iomanager.feature.feature.group
         com.broadhop.notifications.service.feature.feature.group
         com.broadhop.server.runtime.product
         com.broadhop.snmp.feature.feature.group
Creating MD5 Checksum... /var/qps/images/iomanager01.tar.gz.md5chksum
Building image: /var/qps/images/iomanager02.tar.gz
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.iomanager.feature.feature.group
         com.broadhop.notifications.service.feature.feature.group
         com.broadhop.server.runtime.product
         com.broadhop.snmp.feature.feature.group
Creating MD5 Checksum... /var/qps/images/iomanager02.tar.gz.md5chksum
Building image: /var/qps/images/pb.tar.gz
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.client.feature.audit.feature.group
         com.broadhop.client.feature.balance.feature.group
         com.broadhop.client.feature.custrefdata.feature.group
         com.broadhop.client.feature.diameter2.feature.group
         com.broadhop.client.feature.notifications.feature.group
         com.broadhop.client.feature.spr.feature.group
         com.broadhop.client.feature.unifiedapi.feature.group
         com.broadhop.client.feature.vouchers.feature.group
         com.broadhop.client.feature.ws.feature.group
         com.broadhop.client.product
Creating MD5 Checksum... /var/qps/images/pb.tar.gz.md5chksum
Building image: /var/qps/images/pcrf.tar.gz
```

```
Installing from:
         file:///var/qps/.tmp/release
Installing features:
         com.broadhop.audit.service.feature.feature.group
         com.broadhop.balance.service.feature.feature.group
         com.broadhop.balance.spr.feature.feature.group
         com.broadhop.custrefdata.service.feature.feature.group
         com.broadhop.diameter2.local.feature.feature.group
         com.broadhop.externaldatacache.memcache.feature.feature.group
         com.broadhop.notifications.local.feature.feature.group
         com.broadhop.policy.feature.feature.group
         com.broadhop.server.runtime.product
         com.broadhop.snmp.feature.feature.group
         com.broadhop.spr.dao.mongo.feature.feature.group
         com.broadhop.spr.feature.feature.group
         com.broadhop.ui.controlcenter.feature.feature.group
         com.broadhop.unifiedapi.interface.feature.feature.group
         com.broadhop.unifiedapi.ws.service.feature.feature.group
         com.broadhop.vouchers.service.feature.feature.group
         com.broadhop.ws.service.feature.feature.group
Creating MD5 Checksum... /var/qps/images/pcrf.tar.gz.md5chksum
Copying portal default database to /var/qps/images/portal_dump.tar.gz
Creating MD5 Checksum for portal dump...
Copying portal to /var/qps/images/portal.tar.gz
Creating MD5 Checksum for portal.tar.gz...
Copying wispr.war to /var/qps/images/wispr.war
Output images to /var/qps/images/
```

build_etc.sh

This command is executed from Cluster Manager to rebuild etc.tar.gz in /etc/broadhop/ directory.

Syntax

```
/var/qps/install/current/scripts/build/build_etc.sh
```

Executable on VMs

Cluster Manager

Example

```
[root@host /]# /var/qps/install/current/scripts/build/build_etc.sh
Building /etc/broadhop...
Copying to /var/qps/images/etc.tar.gz...
Creating MD5 Checksum...
```

build_set.sh

This command is used to rebuild replica sets. This command is normally only run the first time the environment starts, but can be used if CPS databases must be rebuilt.

Syntax

```
/var/qps/bin/support/mongo/build_set.sh [--help]
```

Executable on VMs

All

Example

To create replica-sets for SPR:

```
[root@host /]# /var/qps/bin/support/mongo/build_set.sh --spr --create
Starting Replica-Set Creation
Please select your choice: replica sets sharded (1) or non-sharded (2):
2
```

capture_env.sh

This command collects most of the debug logs to debug an issue.

Syntax

```
/var/qps/bin/support/env/capture_env.sh
```

Executable on VMs

pcrfclient01/02

Output

This command provides the following information to collect logs:

- -h|--help: Show usage
- -q|--qns: For capturing qns logs (default is to skip qns logs)
- -t|--trap: For capturing trap logs (default is to skip trap logs)
- -m|--mongo: For capturing mongo logs (default is to skip mongo logs)
- -v|--var-log: For capturing /var/log/messages (default is to skip the log)
- -a|--age: Should be followed by maximum age of log based on last modification time (defaults to 1 day)
- -n|--host: Should be followed by common separated list of hostnames for capturing logs (defaults to all hosts)

Example

```
[root@host /]# /var/qps/bin/support/env/capture_env.sh
Creating archive of QPS environment information...
--------------------------
Capturing /etc/broadhop...
Capturing logs...
Capturing Policy Builder data...
Capturing installed software versions...
```

change_passwd.sh

Change the Control Center user's (Linux user) password on Cluster Manager VM or OAM (pcrfclient) VM.

Syntax

```
/var/qps/bin/support/change_passwd.sh [-h]
```

Executable on VMs

All

Example

```
Enter username whose password needs to be changed:
Enter new password:
Re-enter new password:

Done.
Disconnecting from pcrfclient01... done.
```

cleanup_license.sh

Cleans up the records related to license in the licensedfeats collection in the sharding database. This command must be run as root user when license file is updated on the OAM (pcrfclient) machine.

Syntax

```
/var/qps/bin/support/mongo/cleanup_license.sh [-h]
```

Executable on VMs

- Cluster Manager
- pcrfclient01/02

component_alarm_reports.py

This command is used to store or retrieve the open/active component alarms in CPS.

- For clear alarms, it removes the alarms matching the clear alarm.
- For active alarms, it clears old alarms if any and adds the latest alarm.

Syntax

```
component_alarm_reports.py -h
usage: component_alarm_reports.py [-h] --action {update,report}
                                  [--eventhost EVENTHOST] [--date DATE]
                                  [--name NAME] [--facility FACILITY]
                                  [--severity SEVERITY] [--info INFO]
```

```
CPS Update/Report Component Alarm(s) to/from Mongo DB
optional arguments:
  -h, --help            show this help message and exit
  --action {update,report}, -a {update,report}
                        Action value update : Update an alarm. report : Report
                        active alarms
  --eventhost EVENTHOST, -e EVENTHOST
                        Event Host Name
  --date DATE, -d DATE  Date of event
  --name NAME, -n NAME  Name of alarm
  --facility FACILITY, -f FACILITY
                        Facility of alarm
  --severity SEVERITY, -s SEVERITY
                        Severity of alarm
  --info INFO, -i INFO  Info of alarm
```

Attention The `--action update` parameter is for Cisco Internal Use Only.

Path:

On Cluster Manager: `/var/qps/install/current/scripts/modules/component_alarm_reports.py`

On pcrfclient and policy director VMs:
`/var/qps/bin/install/current/scripts/modules/component_alarm_reports.py`

Executable on VMs

Cluster Manager, Policy Director and OAM (pcrfclient) nodes

Examples

To retrieve the active alarms:

```
component_alarm_reports.py -a report
event_host=lb02 name=ProcessDown severity=critical facility=operatingsystem
date=2017-22-11,10:13:49,310329511,+00:00 info=corosync process is down
```

copytoall.sh

Prior to 7.0.5 release, in order to propagate the changes done in Cluster Manager, user used to execute `reinit.sh` which in turn triggers each CPS VM to download and install the updated VM images from the Cluster Manager and it time consuming process.

In CPS 7.0.5 and higher releases, if minor changes are made to any file in Cluster Manager, instead of executing `reinit.sh` script, use this command to synchronize the modified files from Cluster Manager to all other VMs.

Syntax

```
copytoall.sh
```

Executable on VMs

Cluster Manager

Note In case executing `copytoall.sh` command from qns-admin, prefix sudo before the command.

Example

1. If the user updated `/etc/broadhop/logback.xml` file in Cluster Manager.
2. Build etc directory on each cluster by executing `build_all.sh` from Cluster Manager to rebuild CPS package script.

   ```
   /var/qps/install/current/scripts/build_all.sh
   ```

3. Execute the following command to copy the file:

   ```
   SSHUSER_PREFERROOT=true copytoall.sh /etc/broadhop/logback.xml /etc/broadhop/logback.xml
   ```

diagnostics.sh

Runs a set of diagnostics and displays the current state of the system. If any components are not running, red failure messages are displayed.

Note RADIUS-based policy control is no longer supported in CPS 14.0.0 and later releases as 3GPP Gx Diameter interface has become the industry-standard policy control interface.

Syntax

```
/var/qps/bin/diag/diagnostics.sh -h
Usage: /var/qps/bin/diag/diagnostics.sh [options]
This script runs checks (i.e. diagnostics) against the various access, monitoring, and
configuration points of a running CPS system.
In HA/GR environments, the script always does a ping check for all VMs prior to any other
checks and adds any that fail the ping test to the IGNORED_HOSTS variable. This helps reduce
 the possibility for script function errors.
NOTE: See /var/qps/bin/diag/diagnostics.ini to disable certain checks for the HA/GR env
persistently. The use of a flag will override the diagnostics.ini value.
Examples:
    /var/qps/bin/diag/diagnostics.sh -q
    /var/qps/bin/diag/diagnostics.sh --basic_ports --clock_skew -v
--ignored_hosts='portal01,portal02'

Options:
    --basic_ports : Run basic port checks
        For AIO: 80, 11211, 27017, 27749, 7070, 8080, 8090, 8182, 9091, 9092
        For HA/GR: 80, 11211, 7070, 8080, 8081, 8090, 8182, 9091, 9092, and Mongo DB ports
 based on /etc/broadhop/mongoConfig.cfg
    --clock_skew : Check clock skew between lb01 and all vms (Multi-Node Environment only)
    --diskspace : Check diskspace
    --get_active_alarms : Get the active alarms in the CPS
    --get_replica_status : Get the status of the replica-sets present in environment.
(Multi-Node Environment only)
```

```
   --get_sharding_status : Get the status of the sharding information present in environment.
 (Multi-Node Environment only)
    --get_shard_health : Get the status of the sharded database information present in
environment. (Multi-Node Environment only)
    --get_peer_status: Get the diameter peers present in the environemt.
    --get_sharded_replica_status : Get the status of the shards present in environment.
(Multi-Node Environment only)
   --ha_proxy : Connect to HAProxy to check operation and performance statistics, and ports
 (Multi-Node Environment only)
        http://lbvip01:5540/haproxy?stats
        http://lbvip01:5540//haproxy-diam?stats
    --help -h : Help - displays this help
    --hostnames : Check hostnames are valid (no underscores, resolvable, in
/etc/broadhop/servers) (AIO only)
    --ignored_hosts : Ignore the comma separated list of hosts. For example
--ignored_hosts='portal01,portal02'
        Default is 'portal01,portal02,portallb01,portallb02' (Multi-Node Environment only)
    --ping_check : Check ping status for all VM
    --qns_diagnostics : Retrieve diagnostics from CPS java processes
    --qns_login : Check qns user passwordless login
    --quiet -q : Quiet output - display only failed diagnostics
    --radius : Run radius specific checks
    --redis  : Run redis specific checks
   --svn : Check svn sync status between pcrfclient01 & pcrfclient02 (Multi-Node Environment
 only)
    --tacacs : Check Tacacs server reachability
    --swapspace : Check swap space
   --verbose -v : Verbose output - display *all* diagnostics (by default, some are grouped
 for readability)
    --virtual_ips : Ensure Virtual IP Addresses are operational (Multi-Node Environment
only)
    --vm_allocation : Ensure VM Memory and CPUs have been allocated according to
recommendations
```

The test for swap memory usage must have the following criteria :

- The test passes if the swap space used is less than 200 MB.
- The script issues a warning if the swap space used is between 200 MB and 1000 MB.
- The status fails if the swap memory used exceeds 1000 MB.

Executable on VMs

Cluster Manager and OAM (pcrfclient) nodes

Example

```
[root@pcrfclient01 ~]# diagnostics.sh
QNS Diagnostics
Checking basic ports (80, 7070, 27017, 27717-27720, 27749, 8080, 9091)...[PASS]
Checking qns passwordless logins on all boxes...[PASS]
Validating hostnames...[PASS]
Checking disk space for all VMs...[PASS]
Checking swap space for all VMs...[PASS]
Checking for clock skew...[PASS]
Retrieving QNS diagnostics from qns01:9045...[PASS]
Retrieving QNS diagnostics from qns02:9045...[PASS]
Checking HAProxy status...[PASS]
Checking VM CPU and memory allocation for all VMs...[PASS]
Checking Virtual IPs are up...[PASS]
[root@pcrfclient01 ~]#
```

List of Active Alarms

To get the list of active alarms, execute the `diagnostics.sh --get_active_alarms` command. Here is a sample output:

```
#diagnostics.sh --get_active_alarms

CPS Diagnostics HA Multi-Node Environment
--------------------------
Active Application Alarm Status
-------------------------------------------------------------------------------
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,
10:47:34,051+0000  msg="3001:Host: site-host-gx Realm: site-gx-client.com is down"
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,
10:47:34,048+0000  msg="3001:Host: site-host-sd Realm: site-sd-client.com is down"
id=1000 sub_id=3001  event_host=lb01  status=down date=2017-11-22,
10:45:17,927+0000  msg="3001:Host: site-server Realm: site-server.com is down"
id=1000 sub_id=3001  event_host=lb02  status=down date=2017-11-22,
10:47:34,091+0000  msg="3001:Host: site-host-rx Realm: site-rx-client.com is down"
id=1000 sub_id=3002  event_host=lb02  status=down date=2017-11-22,
10:47:34,111+0000  msg="3002:Realm: site-server.com:applicationId: 7:all peers are down"
Active Component Alarm Status
-------------------------------------------------------------------------------
event_host=lb02 name=ProcessDown severity=critical facility=operatingsystem
date=2017-22-11,10:13:49,310329511,+00:00 info=corosync process is down
```

Attention

- Due to the limitation of architecture of the CPS SNMP implementation, if the SNMP deamon or policy server (QNS) process on pcrfclient VM restarts, there can be gap between active alarms displayed by the `diagnostics.sh` and active alarms in NMS.
- The date printed for application alarm status is when the alarm was seen at pcrfclient VM. The time for the alarm at NMS is the time before the alarm is received from Policy Director (LB) VM. So there can be a difference in the dates for the same alarm reported in `diagnostics.sh` and in NMS.

The following table list the type of SNMP alarms:

Table 20: IDs - Type of SNMP Alarms

Alarm ID	Type
1000	Application Alarm
7100	Database Alarm
7200	Failover Alarm
7300	Process Alarm
7400	VM Alarm
7700	GR Alarm

For more information on SNMP alarms, refer to *CPS SNMP, Alarms and Clearing Procedures Guide*.

Sample Output of --get_sharding_status

```
-
-------------------------------------------------------------------------------
|-----------------------------------------------------------------------------------------|
|                                    MONGODB SHARDING STATUS INFORMATION
            Date : 2017-12-20 19:02:38 |
|-----------------------------------------------------------------------------------------|


Shard Id    Mongo DB                              State     Backup DB    Removed    Session
 Count

1           sessionmgr01:27717/session_cache      online    false        false      0
2           sessionmgr01:27717/session_cache_2    online    false        false      0
4           sessionmgr01:27717/session_cache_4    online    false        false      0

Rebalance Status: Rebalanced


|-----------------------------------------------------------------------------------------|


Shard Id    Mongo DB                            State     Backup DB    Removed    Session
Count

1           sessionmgr01:37717/session_cache    online    false        false      0

Rebalance Status: Rebalanced
```

dump_utility.py

This collection utility is used to collect standard information from the CPS system in case of issues (system, application, database). This utility collects such information from VM, depending on type of information and VMs selected in the input.

This utility can be executed from anywhere from the terminal. Logs are printed on terminal and written to a log file: `/var/tmp/dumputility-<date_time_when_executed>.log`.

Important Warning messages related to the files that does not exist in the system will not be displayed on the terminal but will be logged only to the log file (`/var/tmp/dumputility-<date_time_when_executed>.log`).

Caution Running the dump utility can be CPU intensive.

Important The dump utility should be run from the Cluster Manager wherever possible.

The following types of information can be collected:

- **Common Information:** This information is common for all type of issues. Information is collected from pcrfclient01 VM. If pcrfclient01 is down, information is collected from pcrfclient02 VM. If both VMs are down, information is collected from Cluster Manager VM. The following information can be fetched:
  - `about.sh` output
  - `diagnostics.sh` output
  - `list_installed_features.sh` output
  - Facter output
  - Consolidated logs
  - Bulkstats files
  - SVN dump from PB config
- **System Information:** This information is useful in troubleshooting system related issues. The following information can be fetched:
  - `sysctl -a` output
  - Information about processes running
  - Firewall configuration
  - Netstat statistics
  - Complete lsof output
  - Total number of open files
  - `ifconfig` output
  - Routing table information
  - Disk usage
  - Monit status
  - Monit summary
  - System logs
  - Sar logs
  - Dmesg logs
  - Secure logs
  - Yum logs
  - Whisper logs
  - Puppet logs
- **Application Information:** This information is useful in troubleshooting application-related issues. The following information can be fetched:
  - Contents of `/etc/broadhop` directory

- `/var/log/broadhop` logs
- monit status
- monit summary

- **Database Information:** This information is useful in troubleshooting database related issues. The following information can be fetched:
  - MongoDB logs
  - Mongostat output
  - `rs.status()` output
  - `rs.conf()` output
  - `/var/qps/bin/support/mongo/session_cache_ops.sh -count` output
  - `top_qps.sh` output for 10 seconds
  - `mongotop` output for 3 seconds
- **OAM (PCRFCLIENT) Specific Data:** The following information can be fetched from OAM (pcrfclient) VMs:
  - carbon logs
  - httpd logs
  - `pcs resource show` output
- **Policy Director (lb) Specific Data:** The following information can be fetched from policy director (load balancer) VMs:
  - SNMP trap logs
  - HAproxy logs
  - `pcs resource show` output
- **Policy Server (QNS) Specific Data:** The following information can be fetched from policy server (QNS) VMs:
  - Thread level CPU/memory usage of java process
  - `jstack` output of java process
  - Policy Server (QNS) logs
  - Policy Server service logs

Syntax

```
dump_utility.py
```

The following options are supported:

- `-v,--vm-type`: Specifies type of VM or single VM name from which information has to be fetched. Multiple VMs are separated by colon. For example, `--vm-type qns:sessionmgr01`.
- `-i,--info-type`: Specifies type of information to be collected. Possible values are application, db, system, vm_specific. Multiple values are separated by colon. For example, `--info-type application:system`.
- `-o,--output-file-name`: Name of the tar file to store fetched information.
- `-h,--help`: Displays help.

Executable on VMs

- Cluster Manager
- pcrfclient01/02

Example

- To fetch system information from Policy Director (lb) VMs:

  ```
  dump_utility.py --info-type system --vm-type lb
  ```

- To fetch application and VM specific information from qns01:

  ```
  dump_utility.py --info-type application:vm_specific --vm-type qns01
  ```

  OR

  ```
  dump_utility.py --info-type application:vm_specific --vm-type sav-qns01
  ```

  where, *sav-qns01* is hostname of qns01 VM.

- To fetch database specific information from all replica sets:

  ```
  dump_utility.py --info-type db --vm-type pcrfclient:sessionmgr
  ```

Sample output:

```
dump_utility.py --info-type application --vm-type sav-qns01
Logs are also getting stored in /var/tmp/dumputility-07-06-2016-04-07-47.log
**********************************************
Collecting information, please wait...
**********************************************
Fetching common information like about.sh/list_installed_features/diagnostics etc from
pcrfclient01
This step takes time, please wait...
Fetching command outputs from pcrfclient01
Fetching files hosts file from pcrfclient01
Fetching files consolidated logs from pcrfclient01
Fetching files Bulkstats file from pcrfclient01
Fetching command outputs from qns01
Fetching files Broadhop dir from qns01
Fetching files Broadhop logs from qns01
**********************************************
Information is collected at  : /var/tmp/07-06-2016-04-07-47.tar.gz
**********************************************
Disconnecting from pcrfclient01... done.
```

Important For non-root users, certain CPS scripts (`about.sh`, `diagnostics.sh` and so on) expects sudo password. For such scripts, output is displayed on terminal and is saved in the file. Also for some data which can only be accessed by root user, permission denied related warning is displayed.

list_installed_features.sh

Displays the features and versions of the features that are installed on each VM in the environment.

Syntax

```
/var/qps/bin/diag/list_installed_features.sh
```

Executable on VMs

All

Example

```
[root@host /]# /var/qps/bin/diag/list_installed_features.sh
Features installed on lb01:9045
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.iomanager.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
Features installed on lb02:9045
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.iomanager.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
Features installed on qns01:9045
 com.broadhop.balance.service.feature=3.4.2.r071203
 com.broadhop.balance.spr.feature=3.4.2.r071203
 com.broadhop.custrefdata.service.feature=2.4.2.r072158
 com.broadhop.diameter2.local.feature=3.4.2.r072694
 com.broadhop.externaldatacache.memcache.feature=7.0.2.r072627
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.policy.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
 com.broadhop.spr.dao.mongo.feature=2.3.2.r071887
 com.broadhop.spr.feature=2.3.2.r071887
 com.broadhop.ui.controlcenter.feature=3.4.2.r070445
 com.broadhop.unifiedapi.interface.feature=2.3.2.r072695
 com.broadhop.unifiedapi.ws.service.feature=2.3.2.r072695
 com.broadhop.vouchers.service.feature=3.4.2.r071203
 com.broadhop.ws.service.feature=1.5.2.r071537
Features installed on qns02:9045
 com.broadhop.balance.service.feature=3.4.2.r071203
 com.broadhop.balance.spr.feature=3.4.2.r071203
 com.broadhop.custrefdata.service.feature=2.4.2.r072158
 com.broadhop.diameter2.local.feature=3.4.2.r072694
 com.broadhop.externaldatacache.memcache.feature=7.0.2.r072627
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.policy.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
```

```
 com.broadhop.snmp.feature=7.0.2.r072627
 com.broadhop.spr.dao.mongo.feature=2.3.2.r071887
 com.broadhop.spr.feature=2.3.2.r071887
 com.broadhop.ui.controlcenter.feature=3.4.2.r070445
 com.broadhop.unifiedapi.interface.feature=2.3.2.r072695
 com.broadhop.unifiedapi.ws.service.feature=2.3.2.r072695
 com.broadhop.vouchers.service.feature=3.4.2.r071203
 com.broadhop.ws.service.feature=1.5.2.r071537
Features installed on qns03:9045
 com.broadhop.balance.service.feature=3.4.2.r071203
 com.broadhop.balance.spr.feature=3.4.2.r071203
 com.broadhop.custrefdata.service.feature=2.4.2.r072158
 com.broadhop.diameter2.local.feature=3.4.2.r072694
 com.broadhop.externaldatacache.memcache.feature=7.0.2.r072627
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.policy.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
 com.broadhop.spr.dao.mongo.feature=2.3.2.r071887
 com.broadhop.spr.feature=2.3.2.r071887
 com.broadhop.ui.controlcenter.feature=3.4.2.r070445
 com.broadhop.unifiedapi.interface.feature=2.3.2.r072695
 com.broadhop.unifiedapi.ws.service.feature=2.3.2.r072695
 com.broadhop.vouchers.service.feature=3.4.2.r071203
 com.broadhop.ws.service.feature=1.5.2.r071537
Features installed on qns04:9045
 com.broadhop.balance.service.feature=3.4.2.r071203
 com.broadhop.balance.spr.feature=3.4.2.r071203
 com.broadhop.custrefdata.service.feature=2.4.2.r072158
 com.broadhop.diameter2.local.feature=3.4.2.r072694
 com.broadhop.externaldatacache.memcache.feature=7.0.2.r072627
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.policy.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
 com.broadhop.spr.dao.mongo.feature=2.3.2.r071887
 com.broadhop.spr.feature=2.3.2.r071887
 com.broadhop.ui.controlcenter.feature=3.4.2.r070445
 com.broadhop.unifiedapi.interface.feature=2.3.2.r072695
 com.broadhop.unifiedapi.ws.service.feature=2.3.2.r072695
 com.broadhop.vouchers.service.feature=3.4.2.r071203
 com.broadhop.ws.service.feature=1.5.2.r071537
Features installed on pcrfclient01:9045
 com.broadhop.controlcenter.feature=7.0.2.r072627
 com.broadhop.faultmanagement.service.feature=1.0.2.r071534
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
Features installed on pcrfclient02:9045
 com.broadhop.controlcenter.feature=7.0.2.r072627
 com.broadhop.faultmanagement.service.feature=1.0.2.r071534
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.server.runtime.product=7.0.2.r072627
 com.broadhop.snmp.feature=7.0.2.r072627
Features installed on all (combined)
 com.broadhop.balance.service.feature=3.4.2.r071203
 com.broadhop.balance.spr.feature=3.4.2.r071203
 com.broadhop.controlcenter.feature=7.0.2.r072627
 com.broadhop.custrefdata.service.feature=2.4.2.r072158
 com.broadhop.diameter2.local.feature=3.4.2.r072694
 com.broadhop.externaldatacache.memcache.feature=7.0.2.r072627
 com.broadhop.faultmanagement.service.feature=1.0.2.r071534
 com.broadhop.infrastructure.feature=7.0.2.r072627
 com.broadhop.iomanager.feature=7.0.2.r072627
```

```
com.broadhop.policy.feature=7.0.2.r072627
com.broadhop.server.runtime.product=7.0.2.r072627
com.broadhop.snmp.feature=7.0.2.r072627
com.broadhop.spr.dao.mongo.feature=2.3.2.r071887
com.broadhop.spr.feature=2.3.2.r071887
com.broadhop.ui.controlcenter.feature=3.4.2.r070445
com.broadhop.unifiedapi.interface.feature=2.3.2.r072695
com.broadhop.unifiedapi.ws.service.feature=2.3.2.r072695
com.broadhop.vouchers.service.feature=3.4.2.r071203
com.broadhop.ws.service.feature=1.5.2.r071537
```

reinit.sh

This command is executed from Cluster Manager. It SSHs to all the CPS VMs and triggers the `/etc/init.d/vm-init.sh` script on each VM to download all the Puppet scripts, CPS softwares, /etc/hosts files and updates the VM with the new software from Cluster Manager to the VM.

Refer to vm-init.sh, on page 230, to trigger this process for a single VM as opposed to all VMs.

Syntax

```
/var/qps/install/current/scripts/upgrade/reinit.sh
```

Executable on VMs

Cluster Manager

Example

```
[root@host /]# /var/qps/install/current/scripts/upgrade/reinit.sh
Running pupdate on lab
Updating /etc/hosts file from installer VM...
Updating /etc/facter/facts.d/bxb1-lb01...
Updating /etc/puppet from installer VM...
```

restartall.sh

This command is executed from Cluster Manager. It stops and restarts all of the Policy Server (QNS) services on all VMs in the CPS cluster. This command is also executed when new software is installed on VMs.

Refer to restartqns.sh, on page 201 to restart Policy Server (QNS) services on a specific VM as opposed to all VMs.

Syntax

```
/var/qps/bin/control/restartall.sh
```

Executable on VMs

Cluster Manager

Note When executing restartall.sh command from qns-admin, prefix sudo before the command.

Example

```
/var/qps/bin/control/restartall.sh
Currently active LB: lb01

This process will restart all QPS software on the nodes in this order:

lb02 pcrfclient02 qns01 qns02 pcrfclient01 lb01
```

restartqns.sh

This command stops and restarts all Policy Server (QNS) services on the target VM.

Syntax

```
/var/qps/bin/control/restartqns.sh hostname
```

Executable on VMs

Cluster Manager

Note When executing restartqns.sh command from qns-admin, prefix sudo before the command.

Example

```
/var/qps/bin/control/restartqns.sh qns01
/var/qps/bin/control/restartqns.sh pcrfclient01
```

runonall.sh

Executes a command, as provided as an argument, on all of the VMs listed in the servers file. These commands must be run as the CPS user on the remote VMs, or they will fail to execute properly.

Syntax

```
/var/qps/bin/control/runonall.sh <executable command>
```

Executable on VMs

All

Note In case executing runonall.sh command from qns-admin, prefix sudo before the command.

Example

```
/var/qps/bin/control/runonall.sh ntpdate -u
```

service

This command is used to control individual services on each VM.

Syntax

```
service < option > | --status-all | [ service_name [ command | --full-restart ] ]
```

Caution Do not use this command for any services managed by the monit service. Use the monit summary command to view the list of services managed by monit. The list of services managed by monit is different on each CPS VM.

session_cache_ops.sh

This command provides information about, and performs operations on the session database.

Syntax

```
/var/qps/bin/support/mongo/session_cache_ops.sh <Argument1> <Argument2>
<Argument1>: --count or --remove
--count
--remove
--statistics-count
--add-shard
--add-ringset
--db-shrink
<Argument2>: site1 or site2 or site3 ... siten
This argument for GR only, in GR setup user need to pass the site number (site1 or site2
...) as second argument
```

Options

--count

This option prints the count of sessions present in all available session_cache* databases.

The session count is the number of allocated entries in the database for unique subscriber sessions on the network. Each allocated entry may have related nested sub-sessions with other session types such as Sy/Rx.

- A session count is incremented when a Gx CCR-I arrives and an entry (Mongo data structure called a document) is allocated.
- If there are other types of sessions related to that unique subscriber during the life of the Gx session (or Sy/Rx) these are nested within the "document".

- If these "other" types of sessions are terminated, they are removed from the document and those counter types are decremented immediately.
- When the Gx CCR-T arrives it is decremented immediately from the Gx Type count.
- Up to 30 seconds later the unique session entry/document is removed, and the Session Total count is decremented.

The other session types should not be used in validating the number of total sessions as this varies greatly between call models and time. These are simply specific totals drawn from each entry.

It is typical for the session_total_count will be slightly more than the Gx_TYPE count due to the 30 second delay. The reason for this delay is that the entry (document) needs to wait for any other related (nested) sessions to close.

On occasion there may be small variance of data between what the pcrfclient01 and pcrfclient02 report, although they are querying the same database. These are, however, comparable.

The counters processing order is that the total session count is performed first and the detailed session (types) are done second. Slight discrepancies/variance in the numbers may occur.

Example

```
# session_cache_ops.sh --count
Session cache operation script
Tue Dec 22 02:26:49 MST 2015
-----------------------------------------------------
Session Replica-set SESSION-SET1
-----------------------------------------------------
Session Database          : Session Count
-----------------------------------------------------
session_cache            : 14
session_cache_2          : 15
session_cache_3          : 12
session_cache_4          : 10
-----------------------------------------------------
No of Sessions in SET1   : 51
-----------------------------------------------------
Total Number of Sessions  : 51
```

--remove

This option removes sessions from all available session_cache* databases.

Warning You will be prompted to confirm this action after running this command. If you proceed, this will remove existing sessions in the replica-set.

Example

```
# session_cache_ops.sh --remove
Session cache operation script
Tue Dec 22 02:29:42 MST 2015
-----------------------------------------------------------
Session Replica-set SESSION-SET1
-----------------------------------------------------------
WARNING: Continuing will remove existing sessions in
          replica-set : SESSION-SET1
CAUTION: This result into loss of session data
Are you sure you want to continue (y/yes or n/no)? : y
```

```
Removing sessions from session_cache db
connecting to: sessionmgr04:27717/session_cache
WriteResult({ "nRemoved" : 1 })
Remove sessions operation completed on session_cache db.
Removing sessions from session_cache_2 db
connecting to: sessionmgr04:27717/session_cache_2
WriteResult({ "nRemoved" : 0 })
Remove sessions operation completed on session_cache_2 db.
Removing sessions from session_cache_3 db
connecting to: sessionmgr04:27717/session_cache_3
WriteResult({ "nRemoved" : 0 })
Remove sessions operation completed on session_cache_3 db.
Removing sessions from session_cache_4 db
connecting to: sessionmgr04:27717/session_cache_4
WriteResult({ "nRemoved" : 0 })
Remove sessions operation completed on session_cache_4 db.
-------------------------------------------------------
```

--statistics-count

This option prints statistics count of the sessions (types if the session Gx, Rx, and so on) in all available session_cache* databases.

Example

```
# session_cache_ops.sh --statistics-count
Session cache operation script
Tue Dec 22 02:28:38 MST 2015
-----------------------------------------------------
Sessions statistic counter on Genaral
-----------------------------------------------------
  Session Type         : Session Count
-----------------------------------------------------
ADMIN-SET1
  EDR                  : 5
  GX_SCE               : 10
-----------------------------------------------------
```

--add-shard

Adds session shards to the session database, either normal shards or hot standby shards.

Example

```
# session_cache_ops.sh --add-shard
Session cache operation script
Tue Dec 22 02:22:24 MST 2015
            Session Sharding
-----------------------------------------------------------

Select type of session shard  Default      [*]
                              Hot Standby  [ ]

Sessionmgr pairs : sessionmgr01:sessionmgr02:27717

Session shards per pair : 4

Creating Session sharding [ Done ]


-----------------------------------------------------------
Note :
- Press 'y' to select the shard type
- If sharding needed for multiple sessionmgr vms with port
```

```
  please provide sessionmge vm with port separated by ':',
  and pair separated by ','
(Ex: sessionmgr01:sessionmgr02:27717,sessionmgr03:sessionmgr04:27717)
```

Note If sharding is required for GR setup, then add *site_id* at end of sessionmgr VM and port, separated by colon (:) in sessionmgr pairs parameter.

For example, `sessionmgr01:sessionmgr02:27717:SITE1`

--add-ringset

This option adds a new set to the ring.

Example

```
# session_cache_ops.sh --add-ringset
Session cache operation script
 Wed Jun  8 18:23:15 EDT 2016
Session cache operation script: addRingSet
 The progress of this script can be monitored in the following log:
/var/log/broadhop/scripts/session_cache_ops_08062016_182315.log
Note :
 Please provide sessionmgr vm separated by ':' and pair separated by ','

 (Ex HA: sessionmgr01-lab:sessionmgr02-lab)
 (Ex GR: sessionmgr01-site1:sessionmgr02-site1,sessionmgr01-site2:sessionmgr02-site2)
Enter cache servers: sessionmgr01,sessionmgr02
Verifying Qnses processes is running
 Adding set sessionmgr01,sessionmgr02 to ring
 Executing OSGI Command> setSkRingSet 1 4 sessionmgr01:11211,
 Executing OSGI Command> setSkRingSet 1 4 sessionmgr02:11211,
 Executing OSGI Command> rebuildSkRing 1
 Ringset added successfully
```

--db-shrink

This option is used after clean of all sessions from CPS mongo database. It performs a synchronization operation by removing session cache database files and copying data files from primary member. This reduces the database size and compact database files and/or reclaim disk space. Currently, this operation does not support specific to replica-set.

Note This option must be performed in maintenance window (if required in production) and when there is no session data.

Example

```
# session_cache_ops.sh --db-shrink
Session cache operation script
Fri May 13 06:17:42 EDT 2016
-----------------------------------------------------
Session DB Shrink Replica-set
-----------------------------------------------------
CAUTION: This option must performed in maintenance window and no session data
Are you sure you want to continue (y/yes or n/no)? : yes
Verify log /var/log/broadhop/scripts/session_cache_ops_13052016_061742.log
```

```
DB Shrink operation completed successfully for set - SESSION-SET1
    DB File count before Shrink: 36
    DB File count after  Shrink: 16
    DB Size before Shrink: 4.2G
    DB Size after  Shrink: 256M

DB Shrink operation completed successfully for set - SESSION-SET2
    DB File count before Shrink: 28
    DB File count after  Shrink: 8
    DB Size before Shrink: 4.0G
    DB Size after  Shrink: 128M
```

Executable on VMs

pcrfclient01/02

set_priority.sh

This command sets the priorities of replica-sets, and replica-set members for High Availability (HA) or Geo-Redundant (GR) CPS deployments.

By default, priority of mongo databases, replica-sets, and members are set in order (with higher priority) as defined in the Mongo Config (mongoConfig.cfg).

Use the `diagnostics.sh --get_replica_status` command to view the status and current priorities of all databases replica-sets.

Note If a member is shown in an unknown state, it is likely that the member is not accessible from one of other members, mostly an arbiter. In that case, you must go to that member and check its connectivity with other members.

Also, you can login to mongo on that member and check its actual status.

Syntax

```
/var/qps/bin/support/mongo/set_priority.sh
```

The following options are supported:

- **Mandatory Options:**

  ```
  --db <db_name>
       [all|session|spr|admin|balance|report|portal|audit|bindings]
  ```

  The `set_priority --db all` command would set the priority of all replica-sets listed in mongoConfig.cfg in descending order. The member that is listed first in the configuration would be assigned the highest priority.

  The `set_priority --db session` command would set the priority of all replica-sets of db type SESSION. By default, priorities are set in descending order.
- **General Options:**

  ```
  --h [ --help ]                show syntax and usage information for this script
  --version                     show version information of this script
  --asc                         Set priority in ascending order (default is descending)
  ```

```
--dsc                          Set priority in descending order
--priority <0|1000>            Set specific priority
--force [false|true]          forces the new priority to be applied (default is false).
```

Note The `--priority <0|1000>` option is not currently supported. Do not use.

Caution Do not use the `--force` option unless instructed by a Cisco representative. By default, the `set_priority.sh` script will only attempt to set the priorities when all members of a replica set are in a healthy state. The `--force` option can be used when the members are NOT in a healthy state.

- **Specific Replica-set Options:**

  `--replSet <setname>` specifies the replica-set name

  This option enables you to specify priority for a particular replica-set. You must provide the *<setname>*.

- **Geo-Redundancy Options:**

  `--sitename [site1|site2]` specifies the GR site to which the operation applies

  This option enables you to specify a GR site. The mongoConfig.cfg must have relevant start and end tags (like #SITE1_START and #SITE1_END).

Executable on VMs

Cluster Manager

Examples

High Availability Options:

```
set_priority.sh --db all
set_priority.sh --db session
set_priority.sh --db session --asc
set_priority.sh --db session --replSet set01
```

Geo-Redundancy Options:

```
set_priority.sh --db session --replSet set01 --sitename <site1|site2>
set_priority.sh --db session --replSet set01 --sitename <site1|site2>
set_priority.sh --db session --replSet set01 --sitename <site1|site2> --force true
```

startall.sh

This command is executed from Cluster Manager. It starts all Policy Server (QNS) services on all VMs in the CPS cluster. This command is also executed when a new software is installed on VMs.

Refer to startqns.sh, on page 208 to start services on a specific VM as opposed to all VMs.

Syntax

```
/var/qps/bin/control/startall.sh
```

Note When executing startall.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

Cluster Manager

Example

```
/var/qps/bin/control/startall.sh
```

startqns.sh

This command is executed from Cluster Manager. It starts all Policy Server (QNS) services on the specified VM.

Syntax

```
/var/qps/bin/control/startqns.sh hostname
```

Note When executing startqns.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

Cluster Manager

Example

```
/var/qps/bin/control/startqns.sh qns01
/var/qps/bin/control/startqns.sh pcrfclient01
```

statusall.sh

This command displays whether the services managed by monit are stopped or running on all VMs. This script can be executed from Cluster Manager or OAM (pcrfclient).

Syntax

```
/var/qps/bin/control/statusall.sh
```

Note When executing statusall.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

- Cluster Manager
- pcrfclient01/02

Output

For each process or program, the command displays:

- **Status**
  - Running – the process/Program is healthy and running
  - Does not exist – the process id specified in the /var/run/processname-pid does not exist. This is a cause for concern if recurring.
  - Waiting – This is normal for a program /process monitored by monit
  - Status ok – This is normal for a program monitored by monit
- **Monitoring Status**
  - Monitored – The process/program is being monitored
  - Not Monitored – The process/program is not under the control of monit
  - Waiting – A transient state which reports as waiting depending upon when the statusall.sh command is run which internally uses monit status command.

Note For more details, see: https://bitbucket.org/tildeslash/monit/issue/114/.

- **Uptime**

  The number of days, hours, and minutes the process or program has been running.

Example

```
[root@host /]# /var/qps/bin/control/statusall.sh
Executing 'sudo /usr/bin/monit status' on all QNS Servers
The Monit daemon 5.5 uptime: 2h 12m
Process 'snmptrapd'
  status                            Running
  monitoring status                 Monitored
  uptime                            15h 33m
Process 'snmpd'
  status                            Running
  monitoring status                 Monitored
  uptime                            2h 12m
Process 'sessionmgr-27017'
  status                            Running
  monitoring status                 Monitored
  uptime                            15h 33m
Process 'qns-2'
  status                            Running
  monitoring status                 Monitored
  uptime                            15h 33m
```

```
Process 'qns-1'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'memcached'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'logstash'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'elasticsearch'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'collectd'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'carbon-cache'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
Process 'carbon-aggregator'
  status                           Running
  monitoring status                Monitored
  uptime                           15h 33m
System 'lab'
  status                           Running
  monitoring status                Monitored
Connection to 127.0.0.1 closed.
```

stopall.sh

This command is executed from Cluster Manager. It stops the Policy Server (QNS) services on each VMs in the CPS cluster.

Refer to stopqns.sh, on page 211 to stop Policy Server (QNS) services on a specific VM as opposed to all VMs.

Syntax

```
/var/qps/bin/control/stopall.sh
```

Note When executing stopall.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

Cluster Manager

Example

```
/var/qps/bin/control/stopall.sh
```

stopqns.sh

This command is executed from Cluster Manager. It stops all Policy Server (QNS) services on the specified VM.

Syntax

```
/var/qps/bin/control/stopqns.sh hostname
```

Note When executing stopqns.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

Cluster Manager

Example

```
/var/qps/bin/control/stopqns.sh qns01
```

summaryall.sh

This command provides a brief status of the services managed by monit on all VMs in the CPS cluster.

Syntax

```
/var/qps/bin/control/summaryall.sh
```

Note When executing summaryall.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

Cluster Manager

Example

```
/var/qps/bin/control/summaryall.sh
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmptrapd'                 Running
Process 'snmpd'                     Running
Program 'vip_trap'                  Status ok
Program 'gr_site_status_trap'       Status ok
Process 'redis-2'                   Running
Process 'redis-1'                   Running
Process 'redis'                     Running
Process 'qns-4'                     Running
Process 'qns-3'                     Running
```

```
Process 'qns-2'                     Running
Process 'qns-1'                     Running
Process 'corosync'                  Running
File 'monitor-qns-4'                Accessible
File 'monitor-qns-3'                Accessible
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'memcached'                 Running
Process 'irqbalance'                Running
Process 'haproxy-diameter'          Running
Process 'haproxy'                   Running
Process 'cutter'                    Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'lb01'                       Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmptrapd'                 Running
Process 'snmpd'                     Running
Program 'vip_trap'                  Status ok
Program 'gr_site_status_trap'       Status ok
Process 'redis-2'                   Running
Process 'redis-1'                   Running
Process 'redis'                     Running
Process 'qns-4'                     Running
Process 'qns-3'                     Running
Process 'qns-2'                     Running
Process 'qns-1'                     Running
Process 'corosync'                  Running
File 'monitor-qns-4'                Accessible
File 'monitor-qns-3'                Accessible
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'memcached'                 Running
Process 'irqbalance'                Running
Process 'haproxy-diameter'          Running
Process 'haproxy'                   Running
Process 'cutter'                    Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'lb02'                       Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr01'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
```

```
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr02'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns01'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns02'                      Running
The Monit daemon 5.14 uptime: 5d 19h 19m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Program 'kpi_trap'                  Status ok
Program 'db_trap'                   Status ok
Program 'failover_trap'             Status ok
Program 'qps_process_trap'          Status ok
Program 'admin_login_trap'          Status ok
Program 'vm_trap'                   Status ok
Program 'qps_message_trap'          Status ok
Program 'ldap_message_trap'         Status ok
Process 'qns-2'                     Running
Process 'qns-1'                     Running
Process 'corosync'                  Running
Program 'monitor_replica'           Status ok
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'logstash'                  Running
Program 'mon_db_for_lb_failover'    Status ok
Program 'mon_db_for_callmodel'      Status ok
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'carbon-cache'              Running
Process 'carbon-aggregator'         Running
Process 'auditrpms.sh'              Running
System 'pcrfclient01'               Running
The Monit daemon 5.14 uptime: 5d 19h 52m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Program 'kpi_trap'                  Status ok
```

```
Program 'db_trap'                   Status ok
Program 'failover_trap'             Status ok
Program 'qps_process_trap'          Status ok
Program 'admin_login_trap'          Status ok
Program 'vm_trap'                   Status ok
Program 'qps_message_trap'          Status ok
Program 'ldap_message_trap'         Status ok
Process 'qns-2'                     Running
Process 'qns-1'                     Running
Process 'corosync'                  Running
Program 'monitor_replica'           Status ok
File 'monitor-qns-2'                Accessible
File 'monitor-qns-1'                Accessible
Process 'logstash'                  Running
Program 'mon_db_for_lb_failover'    Status ok
Program 'mon_db_for_callmodel'      Status ok
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'carbon-cache'              Running
Process 'carbon-aggregator'         Running
Process 'auditrpms.sh'              Running
System 'pcrfclient02'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns03'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns04'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr03'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
```

```
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr04'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns05'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns06'                      Running
The Monit daemon 5.14 uptime: 6d 10h 52m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns07'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns08'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
```

```
Process 'auditrpms.sh'              Running
System 'qns09'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns10'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns11'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns12'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr05'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr06'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
```

```
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr07'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr08'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr13'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr14'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns13'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr09'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m
```

```
Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns14'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr10'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns15'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr11'               Running
The Monit daemon 5.14 uptime: 6d 10h 52m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns16'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'memcached'                 Running
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
```

```
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'sessionmgr12'               Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns17'                      Running
The Monit daemon 5.14 uptime: 6d 10h 51m

Process 'whisper'                   Running
Process 'snmpd'                     Running
Process 'qns-1'                     Running
File 'monitor-qns-1'                Accessible
Program 'cpu_load_monitor'          Status ok
Program 'cpu_load_trap'             Status ok
Program 'gen_low_mem_trap'          Status ok
Process 'collectd'                  Running
Process 'auditrpms.sh'              Running
System 'qns18'                      Running

● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 29532 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 4919 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 4937 (java)
   CGroup: /system.slice/qns-1.service
           └─4937 /usr/bin/java -server -XX:+PrintGCDetails -XX:+PrintGCTimeStamps
-XX:+PrintGCDateStamps -XX:+PrintTenuringDistribution -XX...
● qns-2.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-2.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 29426 ExecStop=/var/qps/bin/support/startqps stop 2 (code=exited, status=0/SUCCESS)

  Process: 4860 ExecStart=/var/qps/bin/support/startqps start 2 (code=exited,
status=0/SUCCESS)
 Main PID: 4876 (java)
   CGroup: /system.slice/qns-2.service
           └─4876 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSize...
● qns-3.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-3.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 29846 ExecStop=/var/qps/bin/support/startqps stop 3 (code=exited, status=0/SUCCESS)

  Process: 4804 ExecStart=/var/qps/bin/support/startqps start 3 (code=exited,
status=0/SUCCESS)
 Main PID: 4820 (java)
   CGroup: /system.slice/qns-3.service
           └─4820 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSize...
● qns-4.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-4.service; static; vendor preset: disabled)
```

```
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 29697 ExecStop=/var/qps/bin/support/startqps stop 4 (code=exited, status=0/SUCCESS)

  Process: 4749 ExecStart=/var/qps/bin/support/startqps start 4 (code=exited,
status=0/SUCCESS)
 Main PID: 4765 (java)
   CGroup: /system.slice/qns-4.service
           └─4765 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSize...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 17896 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 11745 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 11761 (java)
   CGroup: /system.slice/qns-1.service
           └─11761 /usr/bin/java -server -XX:+PrintGCDetails -XX:+PrintGCTimeStamps
-XX:+PrintGCDateStamps -XX:+PrintTenuringDistribution -X...
● qns-2.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-2.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 16763 ExecStop=/var/qps/bin/support/startqps stop 2 (code=exited, status=0/SUCCESS)

  Process: 11690 ExecStart=/var/qps/bin/support/startqps start 2 (code=exited,
status=0/SUCCESS)
 Main PID: 11706 (java)
   CGroup: /system.slice/qns-2.service
           └─11706 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSiz...
● qns-3.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-3.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 15817 ExecStop=/var/qps/bin/support/startqps stop 3 (code=exited, status=0/SUCCESS)

  Process: 11888 ExecStart=/var/qps/bin/support/startqps start 3 (code=exited,
status=0/SUCCESS)
 Main PID: 11906 (java)
   CGroup: /system.slice/qns-3.service
           └─11906 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSiz...
● qns-4.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-4.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 14704 ExecStop=/var/qps/bin/support/startqps stop 4 (code=exited, status=0/SUCCESS)

  Process: 11812 ExecStart=/var/qps/bin/support/startqps start 4 (code=exited,
status=0/SUCCESS)
 Main PID: 11848 (java)
   CGroup: /system.slice/qns-4.service
           └─11848 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -XX:+TieredCompilation -XX:ReservedCodeCacheSiz...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:49 IST; 4 days ago
  Process: 30239 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 805 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited, status=0/SUCCESS)

 Main PID: 821 (java)
   CGroup: /system.slice/qns-1.service
           └─821 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -Xm...
```

```
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:47 IST; 4 days ago
  Process: 10934 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 13549 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 13565 (java)
   CGroup: /system.slice/qns-1.service
           └─13565 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 9805 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 29166 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 29204 (java)
   CGroup: /system.slice/qns-1.service
          └─29204 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -Xms512m -Xmx1024m -javaagent:/opt/broadhop/qns...
● qns-2.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-2.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 9787 ExecStop=/var/qps/bin/support/startqps stop 2 (code=exited, status=0/SUCCESS)

  Process: 28958 ExecStart=/var/qps/bin/support/startqps start 2 (code=exited,
status=0/SUCCESS)
 Main PID: 29007 (java)
   CGroup: /system.slice/qns-2.service
           └─29007 /usr/java/default/bin/java -server -XX:+UnlockDiagnosticVMOptions
-XX:+UnsyncloadClass -Xms2048m -Xmx2048m -javaagent:/op...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 16610 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 18158 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 18185 (java)
   CGroup: /system.slice/qns-1.service
          └─18185 /usr/bin/java -server -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
 -Xms512m -Xmx1024m -javaagent:/opt/broadhop/qns...
● qns-2.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-2.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 16592 ExecStop=/var/qps/bin/support/startqps stop 2 (code=exited, status=0/SUCCESS)

  Process: 17959 ExecStart=/var/qps/bin/support/startqps start 2 (code=exited,
status=0/SUCCESS)
 Main PID: 17977 (java)
   CGroup: /system.slice/qns-2.service
           └─17977 /usr/java/default/bin/java -server -XX:+UnlockDiagnosticVMOptions
-XX:+UnsyncloadClass -Xms2048m -Xmx2048m -javaagent:/op...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:49 IST; 4 days ago
  Process: 9733 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 12732 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 12748 (java)
   CGroup: /system.slice/qns-1.service
```

```
           └─12748 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:47 IST; 4 days ago
  Process: 8925 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 11537 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 11554 (java)
   CGroup: /system.slice/qns-1.service
           └─11554 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:47 IST; 4 days ago
  Process: 8352 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 11089 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 11111 (java)
   CGroup: /system.slice/qns-1.service
           └─11111 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 6327 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 9157 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 9174 (java)
   CGroup: /system.slice/qns-1.service
           └─9174 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 596 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 3623 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 3639 (java)
   CGroup: /system.slice/qns-1.service
           └─3639 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 6463 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 9160 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 9176 (java)
   CGroup: /system.slice/qns-1.service
           └─9176 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:47 IST; 4 days ago
  Process: 2125 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 4861 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
```

```
 Main PID: 4877 (java)
   CGroup: /system.slice/qns-1.service
           └─4877 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 3601 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 6350 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 6372 (java)
   CGroup: /system.slice/qns-1.service
           └─6372 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 32517 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 3098 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 3114 (java)
   CGroup: /system.slice/qns-1.service
           └─3114 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 6986 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 9788 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 9808 (java)
   CGroup: /system.slice/qns-1.service
           └─9808 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:49 IST; 4 days ago
  Process: 2578 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 5592 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 5609 (java)
   CGroup: /system.slice/qns-1.service
           └─5609 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 6041 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 8728 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 8746 (java)
   CGroup: /system.slice/qns-1.service
           └─8746 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:47 IST; 4 days ago
  Process: 32142 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)
```

```
  Process: 2435 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 2458 (java)
   CGroup: /system.slice/qns-1.service
           └─2458 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 2369 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 5110 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 5129 (java)
   CGroup: /system.slice/qns-1.service
           └─5129 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 2900 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 5889 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 5905 (java)
   CGroup: /system.slice/qns-1.service
           └─5905 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -X...
● qns-1.service - CPS Application Start Script
   Loaded: loaded (/etc/systemd/system/qns-1.service; static; vendor preset: disabled)
   Active: active (running) since Wed 2018-04-18 18:18:46 IST; 4 days ago
  Process: 27907 ExecStop=/var/qps/bin/support/startqps stop 1 (code=exited, status=0/SUCCESS)

  Process: 30489 ExecStart=/var/qps/bin/support/startqps start 1 (code=exited,
status=0/SUCCESS)
 Main PID: 30505 (java)
   CGroup: /system.slice/qns-1.service
           └─30505 /usr/bin/java -XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass
-XX:+TieredCompilation -XX:+DisableExplicitGC -server -...
[root@installer ~]#
```

sync_times.sh

This command synchronizes the time between all CPS VMs.

Syntax

For High Availability deployments:

```
/var/qps/bin/support/sync_times.sh ha
```

For Geographic Redundancy deployments:

```
/var/qps/bin/support/sync_times.sh gr
```

Executable on VMs

Cluster Manager

To check the current clock skew of the system, execute the following command:

```
diagnostics.sh --clock_skew -v
```

The output numbers are in seconds. Refer to the following sample output:

```
CPS Diagnostics Multi-Node Environment
--------------------------
Checking for clock skew...
Clock skew not detected between qns01 and lb01. Skew: 1...[PASS]
Clock skew not detected between qns02 and lb01. Skew: 0...[PASS]
Clock skew not detected between lb01 and lb01. Skew: 0...[PASS]
Clock skew not detected between lb02 and lb01. Skew: 0...[PASS]
Clock skew not detected between sessionmgr01 and lb01. Skew: 0...[PASS]
Clock skew not detected between sessionmgr02 and lb01. Skew: 0...[PASS]
Clock skew not detected between pcrfclient01 and lb01. Skew: 0...[PASS]
Clock skew not detected between pcrfclient02 and lb01. Skew: 0...[PASS]
```

syncconfig.sh

This command is executed to synchronize the changes to the VM nodes. The files in the `/var/qps/current_config/etc/broadhop` are zipped to a file and stored in `/var/www/html`. The Puppet scripts in VM downloads the file to the VM and applies the changes to the VM.

Syntax

```
/var/qps/bin/update/syncconfig.sh
/var/qps/install/currentfolder/scripts/bin/update/syncconfig.sh
```

where, currentfolder is version of the current installation.

For example, for CPS 7.0.5, it is 7.0.5.

```
/var/qps/install/7.0.5/scripts/bin/update/syncconfig.sh
```

Executable on VMs

All

Example

```
[root@host /]# /var/qps/bin/update/syncconfig.sh
Building /etc/broadhop...
Copying to /var/qps/images/etc.tar.gz...
Creating MD5 Checksum...
```

terminatesessions

This utility submits bulk session terminate requests.

Note For fresh installations of CPS 10.1.0, this feature is enabled by default. However, for upgrades from systems prior to CPS 10.1.0, this feature needs to be enabled as follows:

In the `/etc/broadhop/pcrf/features` file, add **`com.broadhop.policy.command.feature`**. For more information, refer to "Customize Features" in the "Deployment" section in *CPS Installation Guide for VMware*

Important To eliminate the impact of TPS and session count in the system, add the following entry in the `/etc/broadhop/qns.conf` file on the Cluster Manager VM:

```
-Ddistribution.blocked.duration=1800000
```

The entry value is in milliseconds, which converts to 30 minutes. The recommended value is multiples of 30 minutes.

After configuring the above values, run the following commands:

```
copytoall.sh /etc/broadhop/qns.conf
stopall.sh
startall.sh
```

Syntax

```
/var/qps/bin/support/command --username <USERNAME> --password <PASSWORD> terminatesessions
--criteria <criteria> [--disable_signaling <y/n - default n>] [--rate <throttling rate -
default 100>]
```

Where,

- --username and --password are the user's Control Center credentials.
- --criteria: Identifies the session. Following are some examples:
  - ALL
  - APN eq SOS
  - APN except SOS
  - IMSIRANGE A-B

Remember For the termination of sessions without any criteria (ALL) and termination of sessions with IMSI range as criteria (IMSIRANGE A-B), CPS must be configured to create sessions with **tags** field having **ImsiKey:imsi:<imsivalue>** as element. If this element is not configured, the command does not terminate sessions for ALL and IMSI range as criteria.

- --disable_signaling: Disables signaling on external interface.
- --rate: Defines the throttling rate.

`/var/qps/bin/support/command terminatesessions -h` shows help related to the command option.

Executable on VMs

pcrfclient01/02

Example

```
/var/qps/bin/support/command -u testuser -p cisco123 terminatesessions -c "ALL" -d y
Do you want to proceed with delete command? [y]|n: y
 deleteBulkSession testuser "ALL" false 100
User is : testuser
Criterion is : ALL
Command Criteria type   : ALL
Command Criteria value   : null
Signalling is set to : false
Rate-Limiter value is set to : 100
CommandId submitted successfully : 1471941788159
```

show

This utility shows the status of the submitted command(s).

Syntax

```
/var/qps/bin/support/command --username <USERNAME> --password <PASSWORD> show [--all <All>]
[--id <ID>]
```

Where,

- --username and --password are the user's Control Center credentials.
- --all: Shows the status of all the command requests submitted.
- --id: Shows the status of the submitted command.

Executable on VMs

pcrfclient01/02

Example

```
/var/qps/bin/support/command -u testuser -p cisco123 show
 getCommands
BulkTerminateCommand(1471492548449)- state: COMPLETED submitted: Thu Aug 18 09:25:48 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
BulkTerminateCommand(1471492739896)- state: COMPLETED submitted: Thu Aug 18 09:28:59 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
BulkTerminateCommand(1471493146320)- state: COMPLETED submitted: Thu Aug 18 09:35:46 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
BulkTerminateCommand(1471494348267)- state: COMPLETED submitted: Thu Aug 18 09:55:48 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
BulkTerminateCommand(1471494588431)- state: COMPLETED submitted: Thu Aug 18 09:59:48 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
```

```
/var/qps/bin/support/command -u testuser -p cisco123 show --id 1471494588431
 getCommand 1471494588431
BulkTerminateCommand(1471494588431)- state: COMPLETED submitted: Thu Aug 18 09:59:48 IST
2016 status:
[Eligible for Deletion = 1, Submitted For Deletion = 1, Not Submitted Due To Later Creation
 = 0]
```

cancel

This utility cancels the further execution of the submitted command.

Syntax

```
/var/qps/bin/support/command --username <USERNAME --password <PASSWORD> cancel --id <ID>
```

Where,

- --username and --password are the user's Control Center credentials.
- --id: ID of the submitted command.

Executable on VMs

pcrfclient01/02

Example

```
/var/qps/bin/support/command -u testuser -p cisco123 cancel --id 1471941788159
Do you want to proceed with cancel command? [y]|n: y
 cancelCommand 1471941788159
Command Already completed: 1471941788159
```

top_qps.sh

This command displays performance statistics of CPS VMs.

Syntax

```
/var/qps/bin/control/top_qps.sh <time>
```

where <time> is the number of seconds for which the statistics are to be captured.

Note When executing top_qps.sh command from qns-admin, prefix sudo before the command.

Executable on VMs

pcrfclient01/02

Output

- Average time in ms.

- Number of message transactions processed during n seconds, where n is an integer value in seconds.
- Transactions per second (TPS) is messages/n.
- Error shows any error occurred during execution on the Policy Server (QNS) VM. It could be database error, authentication failure and so on. Details of the error can be seen in the consolidated engine or in the consolidated Policy Server (QNS) log.
- Times used is how much total time it took to process the message.

Example

Figure 58: Example for top_qps.sh

```
---------------------------------------------------------------------------------------------
Host Detail:
qns03,qns01,qns07,qns06,qns08,qns05,qns04
qns02
Measurement timer: 5    QNS Count: 8
---------------------------------------------------------------------------------------------
   Average    Success        TPS      Error  Time Used                    Messages
   31.3868      31706  6341.2000          0   995.1514             diameter_Gx_CCR-U
   32.4724       4490   898.0000          0   145.8012               diameter_Rx_AAR
   31.1475       4630   926.0000          0   144.2129             diameter_Gx_CCR-I
   29.8733       4697   939.4000          0   140.3147              diameter_Syp_AAA
   31.7974       4037   807.4000         36   128.3662               diameter_Rx_STR
   30.5017       3722   744.4000          0   113.5272             diameter_Gx_CCR-T
   30.1642        415    83.0000          0    12.5181              diameter_Syp_STA
   33.1618        221    44.2000          0     7.3288               diameter_Gx_RAA
   27.6945        120    24.0000          0     3.3233 class com.broadhop.cache.TimerExpired
    6.0920          1     0.2000          0     0.0061               diameter_Rx_ASA
---------------------------------------------------------------------------------------------
   Average    Success        TPS      Error  Time Used                    Actions
   13.6113      82620 16524.0000          0  1124.5640 com.broadhop.locking.impl.LockSessionAction
    4.4117      54757 10951.4000          0   241.5785 com.broadhop.cache.impl.actions.GetSessionAction
    2.2590      50002 10000.4000         36   112.9528 com.broadhop.session.UpdateEntry
    0.9880      54760 10952.0000          0    54.1017 com.broadhop.spr.impl.actions.GetSubscriberActionImpl
    0.8557      36553  7310.6000          0    31.2784 com.broadhop.balance.service.actions.OCSLoadBalanceState
    2.4404       4691   938.2000          0    11.4480 com.broadhop.session.CreateEntry
    0.0674      15322  3064.4000          0     1.0321 com.broadhop.balance.service.actions.OCSGetReservationStatusRequest
    0.0174      54734 10946.8000          0     0.9521 com.broadhop.policyintel.impl.actions.StartPolicyReporting
    0.0138      54758 10951.6000          0     0.7581 com.broadhop.policy.impl.service.AddSubscriberService
    0.0183      32091  6418.2000          0     0.5866         send.diameter_Gx_CCA-U
    0.0284      13374  2674.8000          0     0.3795 diameter.create.remote.session.Syp
    0.0283       8649  1729.8000          0     0.2447          send.diameter_Gx_RAR
    0.0316       4691   938.2000          0     0.1484        send.diameter_Gx_CCA-I
    0.0026      54747 10949.4000          0     0.1407 com.broadhop.policyintel.impl.actions.StopPolicyReporting
    0.0279       4691   938.2000          0     0.1309         send.diameter_Syp_AAR
    0.0007     163853 32770.6000          0     0.1194 com.broadhop.policy.impl.actions.LogMessage
    0.0231       3998   799.6000          0     0.0922         send.diameter_Syp_STR
    0.0197       4540   908.0000          0     0.0896          send.diameter_Rx_AAA
    0.0183       4091   818.2000          0     0.0749          send.diameter_Rx_STA
    0.0166       3776   755.2000          0     0.0626        send.diameter_Gx_CCA-T
    1.0123         15     3.0000          0     0.0152 com.broadhop.session.DeleteEntry
    0.0191        102    20.4000          0     0.0020          send.diameter_Rx_ASR

Fri May 1 01:48:21 IST 2015

*** End-of-Collection ***
```

215265

Diameter Synchronization Message Behavior

Some Diameter messages (like UDR) are synchronous Diameter calls, which means that the Policy Server (QNS) will be waiting for a response after sending the Diameter request.

Response of these Diameter message is not captured in top_qps as those message are not processed in policy engine separately.

Average time 3.3676 shown below is round trip time (from UDR sent to UDA received)

Sample Top_Qns

```
---------------------------------------------------------------------------------------------

   Average    Success        TPS      Error  Time Used                    Messages
    9.7211       2910   727.5000          0    28.2883             diameter_Gx_CCR-I
---------------------------------------------------------------------------------------------

   Average    Success        TPS      Error  Time Used                    Actions
    3.6854       2924   731.0000          0    10.7761
com.broadhop.cache.impl.actions.GetSessionAction
    3.3676       2922   730.5000          0     9.8400      send.sync.diameter_Sh_UDR
    0.7908       2919   729.7500          0     2.3083 com.broadhop.session.CreateEntry
```

```
    0.1981       2924   731.0000          0     0.5793
com.broadhop.locking.impl.LockSessionAction
    0.0480       2919   729.7500          0     0.1400         send.diameter_Gx_CCA-I
    0.0126       2924   731.0000          0     0.0370 diameter.create.remote.session.Sh
```

Average time is not applicable for these response messages. However, number of response messages (UDA) received can be seen from Grafana.

vm-init.sh

This command is executed from the VM nodes from `/etc/init.d`, (starts up automatically if VM reboots too). It downloads all the Puppet script, CPS software, /etc/hosts files and updates the VM with the new software.

This command only updates the software and does not restart the CPS software. The new software will be run only after process restart (for example, by executing `/var/qps/bin/control/restartall.sh` script from Cluster Manager).

Syntax

```
/etc/init.d/vm-init.sh
```

Executable on VMs

Any CPS VM

Example

```
/etc/init.d/vm-init.sh
```