Cisco Policy Suite 6.1 Alarming and SNMP Guide

Version 6.1

August 28, 2014

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide.
Addresses, phone numbers, and fax numbers
are listed on the Cisco website at
www.cisco.com/go/offices.


CONTENTS

Preface

Welcome to the *Cisco Policy Suite 6.1 Alarming and SNMP Guide*.

The Cisco Policy Suite (CPS) is a carrier-grade, policy, and subscriber data management software solution that helps service providers control, monetize, and personalize network service offerings like Wi-Fi and BNG (Broadband Network Gateway).

This document shows how to monitor CPS with operational trending information, and how to manage CPS based on system notifications. Proactive monitoring in this way increases service availability and system usability,

Monitoring and alert notifications are provided via Network Monitoring Solutions (NMS) standard Simple Network Management Protocol (SNMP) methodologies.

This preface covers the following topics:

- Audience

Audience

This guide is best used by these readers:

- Deployment engineers
- Implementation engineers
- Network administrators
- Network engineers
- Network operators
- System administrators

This document assumes a general understanding of network architecture and systems management. Specific knowledge of the SNMP, specifically Version 2c, is required. Installation and initial configuration of CPS is a prerequisite.

Monitoring and Alert Notification

Revised: August 28, 2014,

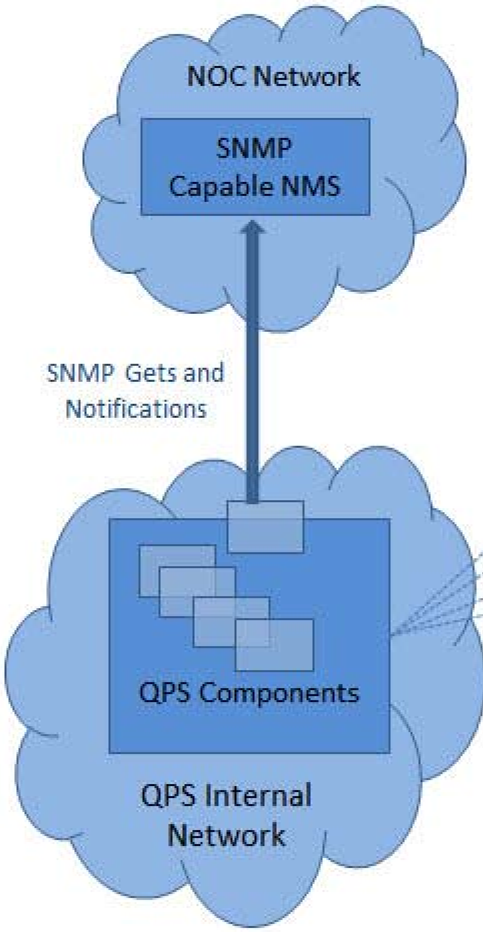
This chapter covers the following sections:

- Architectural Overview
- Technical Architecture
- SNMP System and Application KPIs
- Notifications and Alerting (Traps)
- Configuration and Usage
- Reference Document

Architectural Overview

A CPS deployment is comprised of multiple virtual instances deployed for scaling and high availability purposes. The CPS Systems Monitoring and Notification Alerting system makes the entire CPS installation appear as a single “appliance”. Rather than have administrators deal with a multitude of device agent endpoints, a single entry point (LB) for NMS operational trending and monitoring is used. Likewise, notification alerting from the entire system derives from a single point.

When CPS is deployed in a High Availability (HA) configuration, monitoring and alerting endpoints are deployed as HA as well. This is shown in the illustration below.



Technical Architecture

The Cisco Policy Suite is deployed as a distributed virtual appliance. The standard architecture uses VMWare ESXi virtualization. Multiple physical hardware host components run VMWare ESXi, and each host runs several virtual machines. Within each virtual machine, one-to-many internal CPS components can run. If you add HA capabilities to the deployment, monitoring each CPS component individually becomes unwieldy. The CPS monitoring and alert notification infrastructure simplifies the virtual, physical, and redundant aspects of the architecture.

This section covers the following topics:

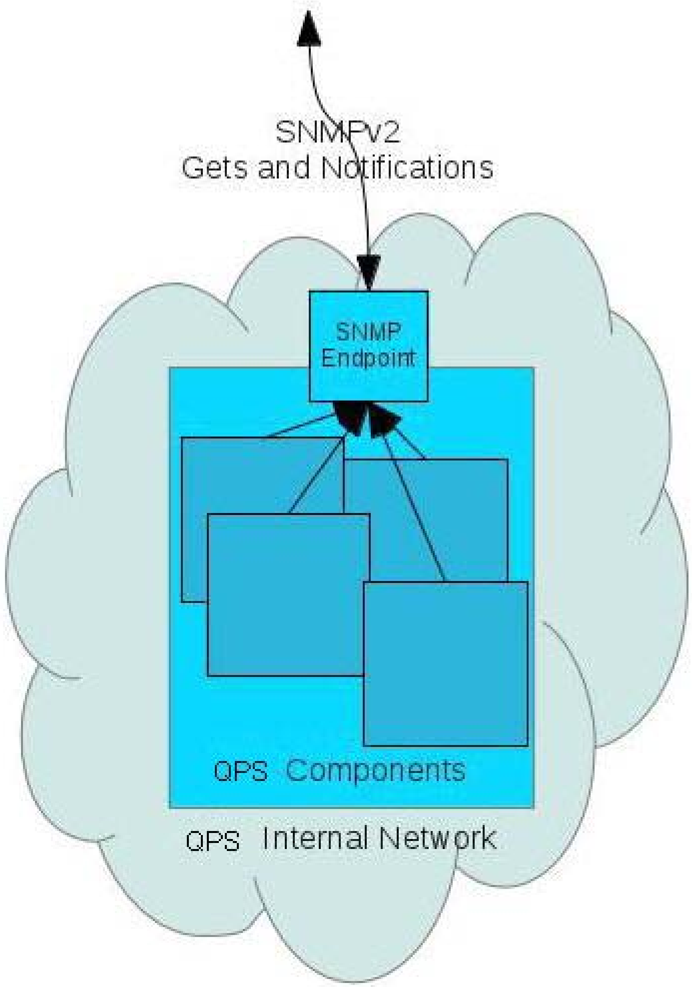
- Protocols and Query Endpoints
- SNMP Object Identifier and Management Information Base
- SNMPv2 Data and Notifications
- Facility
- Severity
- Categorization

- Emergency Severity Note

Protocols and Query Endpoints

The CPS monitoring and alert notification infrastructure provides a simple, standards-based interface for network administrators and NMS. SNMPv2 is the underlying protocol for all monitoring and alert notifications. Standard SNMPv2 gets and notifications (traps) are used throughout the infrastructure and aggregated to an SNMP proxy. This proxy provides a common endpoint for SNMP queries and also maps components into the Cisco Object Identifier (OID) tree structure.

The following drawing shows the aggregation and mapping on the SNMP endpoint (LB).



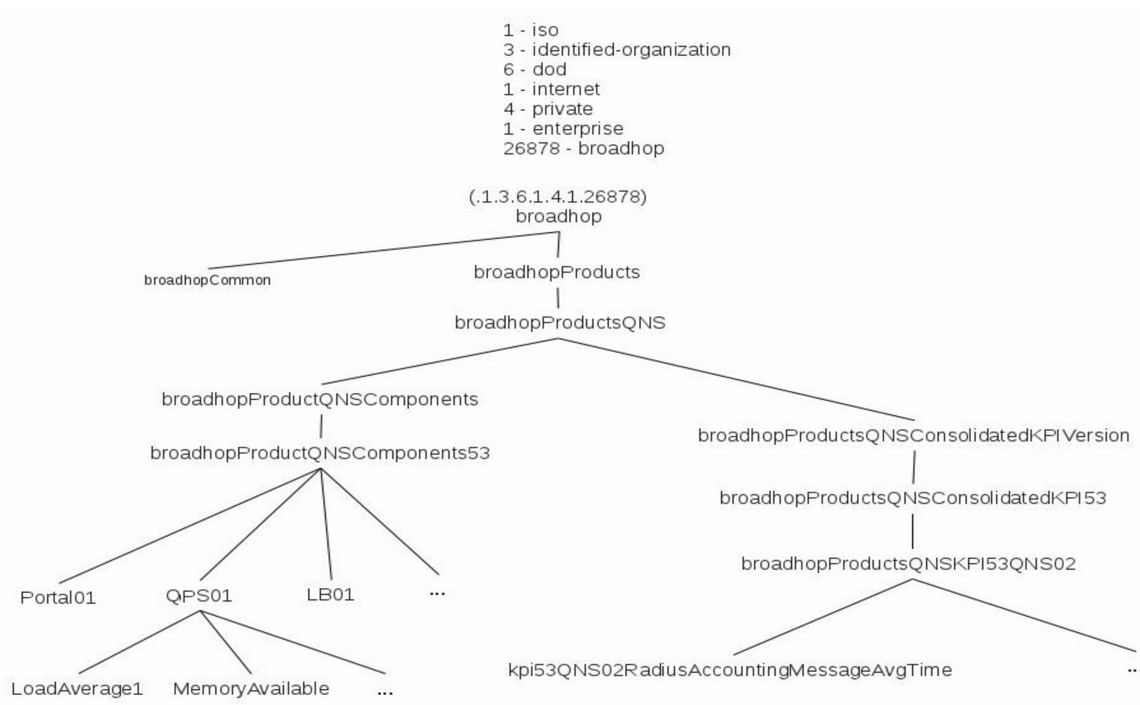
SNMP Object Identifier and Management Information Base

Cisco has a registered private enterprise Object Identifier (OID) of 26878. This OID is the base from which all aggregated CPS metrics are exposed at the SNMP endpoint. The Cisco OID is fully specified and made human-readable through a set of Cisco Management Information Base (MIB-II) files.

The current MIBs are defined as follows:

MIB Filename	Purpose
BROADHOP-MIB.mib	Defines the main structure, including structures and codes.
BROADHOP-QNS-MIB.mib	Defines the retrievable statistics and KPI.
BROADHOP-NOTIFICATION-MIB.mib	Defines Notifications/Traps available.

A graphical overview of the Cisco CPS OID and MIB structure is shown in the next figure.



Note that in the above illustration the entire tree is not shown. The most detailed section of the tree is for pcrfclient.pcrfclient01.oscomponents. More details exist under oscomponents. Similarly, more details exist under other core and peripheral components.

SNMPv2 Data and Notifications

The Monitoring and Alert Notification infrastructure provides standard SNMPv2 get and getnext access to the CPS system. This provides access to targeted metrics to trend and view Key Performance Indicators (KPI). Metrics available through this part of the infrastructure are as general as component load and as specific as transactions processed per second.

SNMPv2 Notifications, in the form of traps (one-way) are also provided by the infrastructure. CPS notifications do not require acknowledgments. These provide both proactive alerts that pre-set thresholds have been passed (for example, Disk is nearing full, CPU load high) and reactive alerting when system components fail or are in a degraded state (for example, DEAD PROCESS, network connectivity outages, etc.).

Notifications and traps are categorized by a methodology similar to UNIX System Logging (syslog) with both Severity and Facility markers. All event notifications (traps) contain these items:

- Facility
- Severity
- Source (device name)
- Device time

These objects enable Network Operations Center (NOC) staff to identify where the issue lies, the Facility (system layer), and the Severity (importance) of the reported issue.

Facility

The generic syslog Facility has the following definitions.

Note Facility defines a system layer starting with physical hardware and progressing to a process running in a particular application.

Number	Facility	Description
0	Hardware	Physical Hardware – Servers, SAN, NIC, Switch, etc.
1	Networking	Connectivity in the OSI (TCP/IP) model.
2	Virtualization	VMWare ESXi (or other) Virtualization
3	Operating System	Linux, Microsoft Windows, etc.
4	Application	Apache httpd, load balancer, CPS sessionmgr, etc.
5	Process	Particular httpd process, Cisco qns01_A, etc.

There may be overlaps in the Facility value as well as gaps if a particular SNMP agent does not have full view into an issue. The Facility reported is always shown as viewed from the reporting SNMP agent.

Severity

In addition to Facility, each notification has a Severity measure. The defined severities are directly from UNIX syslog and defined as follows:

Number	Severity	Description
0	Emergency	System is unusable.
1	Alert	Action must be taken immediately.
2	Critical	Critical conditions.
3	Error	Error conditions.
4	Warning	Warning conditions.
5	Notice	Normal but significant condition.
6	Info	Informational message.
7	Debug	Lower level debug messages.
8	None	Indicates no severity.
9	Clear	The occurred condition has been cleared.

For the purposes of the CPS Monitoring and Alert Notifications system, Severity levels of Notice, Info and Debug are usually not used. Warning conditions are often used for proactive threshold monitoring (for example, Disk usage or CPU Load), which requires some action on the part of administrators, but not immediately. Conversely, Emergency severity indicates that some major component of the system has failed and that either core policy processing, session management or major system function is impacted.

Categorization

Combinations of Facility and Severity create many possibilities of notifications (traps) that might be sent. However, some combinations are more likely than others. The following table lists some noteworthy Facility and Severity categorizations.

Facility.Severity	Categorization	Possibility
Process.Emergency	A single part of an application has dramatically failed.	Possible, but in an HA configuration very unlikely.
Hardware.Debug	A hardware component has sent a debug message.	Possible but highly unlikely.
Operating System.Alert	An Operating System (kernel or resource level) fault has occurred.	Possible as a recoverable kernel fault (on a vNIC for instance).
Application.Emergency	An entire application component has failed.	Unlikely but possible (load balancers failing for instance).
Virtualization.Emergency	The virtualization system has thrown a fault.	Unlikely but possible (VM won't start, or vSwitch fault for instance.

It is not possible quantify every Facility and Severity combination. However, greater experience with CPS leads to better diagnostics. The CPS Monitoring and Alert Notification infrastructure provides a baseline for event definition and notification by an experienced engineer.

Emergency Severity Note

Caution Emergency severities are very, very important! As a general principle, CPS does not throw an Emergency level severity unless the system becomes inaccessible or unusable in some way. An unusable system is extremely rare, but might occur if multiple failures occur in the operating system, virtualization, networking or hardware facilities.

SNMP System and Application KPIs

Many CPS system statistics and Key Performance Indicators (KPI) are available via SNMPv2 gets and walks. Both system device level information and application level information is available. This information is well documented in the BROADHOP-QNS-MIB. A summary of the information available is provided below. This section covers the following topics:

- SNMP System KPIs
- Details of SNMP System KPIs
- SNMP Application KPIs
- Summary of SNMP Application KPIs
- Details of Supported KPIs

SNMP System KPIs

In this table, the system KPI information is provided.

Component	Information
LB01/LB02	CpuUser
PortalLB01/PortalLB02	CpuSystem
PCRFClient01/PCRFClient02	CpuIdle
SessionMgr01/SessionMgr02	LoadAverage1
QNS01/QNS02/QNS03/QNS04	LoadAverage5
Portal01/Portal02	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Details of SNMP System KPIs

The following information is available, and is listed per component. MIB documentation provides units of measure.

```
+--broadhopProductsQNSComponents53(53)
  |
+--broadhopProductsQNSComponents53LB01(11)
  | |
  | +-- -R-- Integer32 component53LB01CpuUser(1)
  | +-- -R-- Integer32 component53LB01CpuSystem(2)
  | +-- -R-- Integer32 component53LB01CpuIdle(3)
  | +-- -R-- Integer32 component53LB01LoadAverage1(4)
  | +-- -R-- Integer32 component53LB01LoadAverage5(5)
  | +-- -R-- Integer32 component53LB01LoadAverage15(6)
  | +-- -R-- Integer32 component53LB01MemoryTotal(7)
  | +-- -R-- Integer32 component53LB01MemoryAvailable(8)
  | +-- -R-- Integer32 component53LB01SwapTotal(9)
  | +-- -R-- Integer32 component53LB01SwapAvailable(10)
  | +-- -R-- Counter   component53LB01Eth0InOctets(11)
  | +-- -R-- Counter   component53LB01Eth0OutOctets(12)
```

```
| +-- -R-- Counter   component53LB01Eth1InOctets(13)
| +-- -R-- Counter   component53LB01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53LB02(12)
| |
| +-- -R-- Integer32 component53LB02CpuUser(1)
| +-- -R-- Integer32 component53LB02CpuSystem(2)
| +-- -R-- Integer32 component53LB02CpuIdle(3)
| +-- -R-- Integer32 component53LB02LoadAverage1(4)
| +-- -R-- Integer32 component53LB02LoadAverage5(5)
| +-- -R-- Integer32 component53LB02LoadAverage15(6)
| +-- -R-- Integer32 component53LB02MemoryTotal(7)
| +-- -R-- Integer32 component53LB02MemoryAvailable(8)
| +-- -R-- Integer32 component53LB02SwapTotal(9)
| +-- -R-- Integer32 component53LB02SwapAvailable(10)
| +-- -R-- Counter   component53LB02Eth0InOctets(11)
| +-- -R-- Counter   component53LB02Eth0OutOctets(12)
| +-- -R-- Counter   component53LB02Eth1InOctets(13)
| +-- -R-- Counter   component53LB02Eth1OutOctets(14)

+--broadhopProductsQNSComponents53PortalLB01(13)
| |
| +-- -R-- Integer32 component53PortalLB01CpuUser(1)
| +-- -R-- Integer32 component53PortalLB01CpuSystem(2)
| +-- -R-- Integer32 component53PortalLB01CpuIdle(3)
| +-- -R-- Integer32 component53PortalLB01LoadAverage1(4)
| +-- -R-- Integer32 component53PortalLB01LoadAverage5(5)
| +-- -R-- Integer32 component53PortalLB01LoadAverage15(6)
| +-- -R-- Integer32 component53PortalLB01MemoryTotal(7)
| +-- -R-- Integer32 component53PortalLB01MemoryAvailable(8)
| +-- -R-- Integer32 component53PortalLB01SwapTotal(9)
| +-- -R-- Integer32 component53PortalLB01SwapAvailable(10)
| +-- -R-- Counter   component53PortalLB01Eth0InOctets(11)
| +-- -R-- Counter   component53PortalLB01Eth0OutOctets(12)
| +-- -R-- Counter   component53PortalLB01Eth1InOctets(13)
| +-- -R-- Counter   component53PortalLB01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53PortalLB02(14)
```

```
  | |
  | +-- -R-- Integer32 component53PortalLB02CpuUser(1)
  | +-- -R-- Integer32 component53PortalLB02CpuSystem(2)
  | +-- -R-- Integer32 component53PortalLB02CpuIdle(3)
  | +-- -R-- Integer32 component53PortalLB02LoadAverage1(4)
  | +-- -R-- Integer32 component53PortalLB02LoadAverage5(5)
  | +-- -R-- Integer32 component53PortalLB02LoadAverage15(6)
  | +-- -R-- Integer32 component53PortalLB02MemoryTotal(7)
  | +-- -R-- Integer32 component53PortalLB02MemoryAvailable(8)
  | +-- -R-- Integer32 component53PortalLB02SwapTotal(9)
  | +-- -R-- Integer32 component53PortalLB02SwapAvailable(10)
  | +-- -R-- Counter   component53PortalLB02Eth0InOctets(11)
  | +-- -R-- Counter   component53PortalLB02Eth0OutOctets(12)
  | +-- -R-- Counter   component53PortalLB02Eth1InOctets(13)
  | +-- -R-- Counter   component53PortalLB02Eth1OutOctets(14)

+--broadhopProductsQNSComponents53PCRFClient01(21)
  | |
  | +-- -R-- Integer32 component53PCRFClient01CpuUser(1)
  | +-- -R-- Integer32 component53PCRFClient01CpuSystem(2)
  | +-- -R-- Integer32 component53PCRFClient01CpuIdle(3)
  | +-- -R-- Integer32 component53PCRFClient01LoadAverage1(4)
  | +-- -R-- Integer32 component53PCRFClient01LoadAverage5(5)
  | +-- -R-- Integer32 component53PCRFClient01LoadAverage15(6)
  | +-- -R-- Integer32 component53PCRFClient01MemoryTotal(7)
  | +-- -R-- Integer32 component53PCRFClient01MemoryAvailable(8)
  | +-- -R-- Integer32 component53PCRFClient01SwapTotal(9)
  | +-- -R-- Integer32 component53PCRFClient01SwapAvailable(10)
  | +-- -R-- Counter   component53PCRFClient01Eth0InOctets(11)
  | +-- -R-- Counter   component53PCRFClient01Eth0OutOctets(12)
  | +-- -R-- Counter   component53PCRFClient01Eth1InOctets(13)
  | +-- -R-- Counter   component53PCRFClient01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53PCRFClient02(22)
  | |
  | +-- -R-- Integer32 component53PCRFClient02CpuUser(1)
  | +-- -R-- Integer32 component53PCRFClient02CpuSystem(2)
  | +-- -R-- Integer32 component53PCRFClient02CpuIdle(3)
```

```
| +-- -R-- Integer32 component53PCRFClient02LoadAverage1(4)
| +-- -R-- Integer32 component53PCRFClient02LoadAverage5(5)
| +-- -R-- Integer32 component53PCRFClient02LoadAverage15(6)
| +-- -R-- Integer32 component53PCRFClient02MemoryTotal(7)
| +-- -R-- Integer32 component53PCRFClient02MemoryAvailable(8)
| +-- -R-- Integer32 component53PCRFClient02SwapTotal(9)
| +-- -R-- Integer32 component53PCRFClient02SwapAvailable(10)
| +-- -R-- Counter   component53PCRFClient02Eth0InOctets(11)
| +-- -R-- Counter   component53PCRFClient02Eth0OutOctets(12)
| +-- -R-- Counter   component53PCRFClient02Eth1InOctets(13)
| +-- -R-- Counter   component53PCRFClient02Eth1OutOctets(14)

+--broadhopProductsQNSComponents53SessionMgr01(31)
| |
| +-- -R-- Integer32 component53SessionMgr01CpuUser(1)
| +-- -R-- Integer32 component53SessionMgr01CpuSystem(2)
| +-- -R-- Integer32 component53SessionMgr01CpuIdle(3)
| +-- -R-- Integer32 component53SessionMgr01LoadAverage1(4)
| +-- -R-- Integer32 component53SessionMgr01LoadAverage5(5)
| +-- -R-- Integer32 component53SessionMgr01LoadAverage15(6)
| +-- -R-- Integer32 component53SessionMgr01MemoryTotal(7)
| +-- -R-- Integer32 component53SessionMgr01MemoryAvailable(8)
| +-- -R-- Integer32 component53SessionMgr01SwapTotal(9)
| +-- -R-- Integer32 component53SessionMgr01SwapAvailable(10)
| +-- -R-- Counter   component53SessionMgr01Eth0InOctets(11)
| +-- -R-- Counter   component53SessionMgr01Eth0OutOctets(12)
| +-- -R-- Counter   component53SessionMgr01Eth1InOctets(13)
| +-- -R-- Counter   component53SessionMgr01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53SessionMgr02(32)
| |
| +-- -R-- Integer32 component53SessionMgr02CpuUser(1)
| +-- -R-- Integer32 component53SessionMgr02CpuSystem(2)
| +-- -R-- Integer32 component53SessionMgr02CpuIdle(3)
| +-- -R-- Integer32 component53SessionMgr02LoadAverage1(4)
| +-- -R-- Integer32 component53SessionMgr02LoadAverage5(5)
| +-- -R-- Integer32 component53SessionMgr02LoadAverage15(6)
| +-- -R-- Integer32 component53SessionMgr02MemoryTotal(7)
```

```
| +-- -R-- Integer32 component53SessionMgr02MemoryAvailable(8)
| +-- -R-- Integer32 component53SessionMgr02SwapTotal(9)
| +-- -R-- Integer32 component53SessionMgr02SwapAvailable(10)
| +-- -R-- Counter   component53SessionMgr02Eth0InOctets(11)
| +-- -R-- Counter   component53SessionMgr02Eth0OutOctets(12)
| +-- -R-- Counter   component53SessionMgr02Eth1InOctets(13)
| +-- -R-- Counter   component53SessionMgr02Eth1OutOctets(14)

+--broadhopProductsQNSComponents53QNS01(41)
| |
| +-- -R-- Integer32 component53QNS01CpuUser(1)
| +-- -R-- Integer32 component53QNS01CpuSystem(2)
| +-- -R-- Integer32 component53QNS01CpuIdle(3)
| +-- -R-- Integer32 component53QNS01LoadAverage1(4)
| +-- -R-- Integer32 component53QNS01LoadAverage5(5)
| +-- -R-- Integer32 component53QNS01LoadAverage15(6)
| +-- -R-- Integer32 component53QNS01MemoryTotal(7)
| +-- -R-- Integer32 component53QNS01MemoryAvailable(8)
| +-- -R-- Integer32 component53QNS01SwapTotal(9)
| +-- -R-- Integer32 component53QNS01SwapAvailable(10)
| +-- -R-- Counter   component53QNS01Eth0InOctets(11)
| +-- -R-- Counter   component53QNS01Eth0OutOctets(12)
| +-- -R-- Counter   component53QNS01Eth1InOctets(13)
| +-- -R-- Counter   component53QNS01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53QNS02(42)
| |
| +-- -R-- Integer32 component53QNS02CpuUser(1)
| +-- -R-- Integer32 component53QNS02CpuSystem(2)
| +-- -R-- Integer32 component53QNS02CpuIdle(3)
| +-- -R-- Integer32 component53QNS02LoadAverage1(4)
| +-- -R-- Integer32 component53QNS02LoadAverage5(5)
| +-- -R-- Integer32 component53QNS02LoadAverage15(6)
| +-- -R-- Integer32 component53QNS02MemoryTotal(7)
| +-- -R-- Integer32 component53QNS02MemoryAvailable(8)
| +-- -R-- Integer32 component53QNS02SwapTotal(9)
| +-- -R-- Integer32 component53QNS02SwapAvailable(10)
| +-- -R-- Counter   component53QNS02Eth0InOctets(11)
```

```
| +-- -R-- Counter   component53QNS02Eth0OutOctets(12)
| +-- -R-- Counter   component53QNS02Eth1InOctets(13)
| +-- -R-- Counter   component53QNS02Eth1OutOctets(14)

+--broadhopProductsQNSComponents53QNS03(43)
| |
| +-- -R-- Integer32 component53QNS03CpuUser(1)
| +-- -R-- Integer32 component53QNS03CpuSystem(2)
| +-- -R-- Integer32 component53QNS03CpuIdle(3)
| +-- -R-- Integer32 component53QNS03LoadAverage1(4)
| +-- -R-- Integer32 component53QNS03LoadAverage5(5)
| +-- -R-- Integer32 component53QNS03LoadAverage15(6)
| +-- -R-- Integer32 component53QNS03MemoryTotal(7)
| +-- -R-- Integer32 component53QNS03MemoryAvailable(8)
| +-- -R-- Integer32 component53QNS03SwapTotal(9)
| +-- -R-- Integer32 component53QNS03SwapAvailable(10)
| +-- -R-- Counter   component53QNS03Eth0InOctets(11)
| +-- -R-- Counter   component53QNS03Eth0OutOctets(12)
| +-- -R-- Counter   component53QNS03Eth1InOctets(13)
| +-- -R-- Counter   component53QNS03Eth1OutOctets(14)

+--broadhopProductsQNSComponents53QNS04(44)
| |
| +-- -R-- Integer32 component53QNS04CpuUser(1)
| +-- -R-- Integer32 component53QNS04CpuSystem(2)
| +-- -R-- Integer32 component53QNS04CpuIdle(3)
| +-- -R-- Integer32 component53QNS04LoadAverage1(4)
| +-- -R-- Integer32 component53QNS04LoadAverage5(5)
| +-- -R-- Integer32 component53QNS04LoadAverage15(6)
| +-- -R-- Integer32 component53QNS04MemoryTotal(7)
| +-- -R-- Integer32 component53QNS04MemoryAvailable(8)
| +-- -R-- Integer32 component53QNS04SwapTotal(9)
| +-- -R-- Integer32 component53QNS04SwapAvailable(10)
| +-- -R-- Counter   component53QNS04Eth0InOctets(11)
| +-- -R-- Counter   component53QNS04Eth0OutOctets(12)
| +-- -R-- Counter   component53QNS04Eth1InOctets(13)
| +-- -R-- Counter   component53QNS04Eth1OutOctets(14)
```

```
+--broadhopProductsQNSComponents53Portal01(51)
  | |
  | +-- -R-- Integer32 component53Portal01CpuUser(1)
  | +-- -R-- Integer32 component53Portal01CpuSystem(2)
  | +-- -R-- Integer32 component53Portal01CpuIdle(3)
  | +-- -R-- Integer32 component53Portal01LoadAverage1(4)
  | +-- -R-- Integer32 component53Portal01LoadAverage5(5)
  | +-- -R-- Integer32 component53Portal01LoadAverage15(6)
  | +-- -R-- Integer32 component53Portal01MemoryTotal(7)
  | +-- -R-- Integer32 component53Portal01MemoryAvailable(8)
  | +-- -R-- Integer32 component53Portal01SwapTotal(9)
  | +-- -R-- Integer32 component53Portal01SwapAvailable(10)
  | +-- -R-- Counter   component53Portal01Eth0InOctets(11)
  | +-- -R-- Counter   component53Portal01Eth0OutOctets(12)
  | +-- -R-- Counter   component53Portal01Eth1InOctets(13)
  | +-- -R-- Counter   component53Portal01Eth1OutOctets(14)

+--broadhopProductsQNSComponents53Portal02(52)
| |
  | +-- -R-- Integer32 component53Portal02CpuUser(1)
  | +-- -R-- Integer32 component53Portal02CpuSystem(2)
  | +-- -R-- Integer32 component53Portal02CpuIdle(3)
  | +-- -R-- Integer32 component53Portal02LoadAverage1(4)
  | +-- -R-- Integer32 component53Portal02LoadAverage5(5)
  | +-- -R-- Integer32 component53Portal02LoadAverage15(6)
  | +-- -R-- Integer32 component53Portal02MemoryTotal(7)
  | +-- -R-- Integer32 component53Portal02MemoryAvailable(8)
  | +-- -R-- Integer32 component53Portal02SwapTotal(9)
  | +-- -R-- Integer32 component53Portal02SwapAvailable(10)
  | +-- -R-- Counter   component53Portal02Eth0InOctets(11)
  | +-- -R-- Counter   component53Portal02Eth0OutOctets(12)
  | +-- -R-- Counter   component53Portal02Eth1InOctets(13)
  | +-- -R-- Counter   component53Portal02Eth1OutOctets(14)
```

SNMP Application KPIs

Current version Key Performance Indicators (KPI) information is available at the OID root of:

.1.3.6.1.4.1.26878.200.2.3.53

This corresponds to an MIB of:

.iso
.identified-organization
.dod
.internet
.private
.enterprise
.broadhop
.broadhopProducts
.broadhopProductsQNS
.broadhopProductsQNSKPIVersion
.broadhopProductsQNSKPI53

Summary of SNMP Application KPIs

The following application KPI's are available for monitoring on each node using SNMP Get and Walk utilities.

Component	Information
LB01/LB02	PCRFProxyExternalCurrentSessions: is the number of open connections to lbvip01:8080. PCRFProxyInternalCurrentSessions: is the number of open connections to lbvip02:8080
PortalLB01/PortalLB02	PortalProxyExternalCurrentSessions: is the number of connections to sslvip01:80.
PCRFClient01/PCRFClient02	----------
SessionMgr01/SessionMgr02	----------

Component	Information
QNS01 QNS02 QNS03 QNS04	PolicyCount: The number of processed policy messages. QueueSize: The number of entries in the processing queue. The default queue size is 500, but this can be configured by the customer in the Policy Builder. You can also see the number of dropped messages in the statistics files. FailedEnqueueCount ErrorCount SessionCount FreeMemory
Portal01 Portal02	----------

Details of Supported KPIs

The following information is available, and are supported in current release. MIB documentation provides units of measure.

```
+--broadhopProductsQNSKPI53(53)
  +--broadhopProductsQNSKPI53LB01(11)
| |
  | +-- -R-- String    kpi53LB01PCRFProxyExternalCurrentSessions(1)
  | |        Textual Convention: DisplayString
  | |        Size: 0..255
  | +-- -R-- String    kpi53LB01PCRFProxyInternalCurrentSessions(2)
  |          Textual Convention: DisplayString
  |          Size: 0..255
  +--broadhopProductsQNSKPI53LB02(12)
  | |
  | +-- -R-- String    kpi53LB02PCRFProxyExternalCurrentSessions(1)
  | |        Textual Convention: DisplayString
  | |        Size: 0..255
  | +-- -R-- String    kpi53LB02PCRFProxyInternalCurrentSessions(2)
  |          Textual Convention: DisplayString
  |          Size: 0..255
  +--broadhopProductsQNSKPI53PortalLB01(13)
  | |
  | +-- -R-- String    kpi53PortalLB01PortalProxyExternalCurrentSessions(1)
```

```
|         Textual Convention: DisplayString
|         Size: 0..255
+--broadhopProductsQNSKPI53PortalLB02(14)
| |
| +-- -R-- String    kpi53PortalLB02PortalProxyExternalCurrentSessions(1)
|         Textual Convention: DisplayString
|         Size: 0..255
+--broadhopProductsQNSKPI53SessionMgr01(31)
+--broadhopProductsQNSKPI53SessionMgr02(32)
+--broadhopProductsQNSKPI53QNS01(41)
| |
| +-- -R-- Integer32 kpi53QNS01PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS01QueueSize(21)
| +-- -R-- Integer32 kpi53QNS01FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS01ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS01SessionCount(24)
| +-- -R-- Integer32 kpi53QNS01FreeMemoryCount(25)

+--broadhopProductsQNSKPI53QNS02(42)
| |
| +-- -R-- Integer32 kpi53QNS02PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS02QueueSize(21)
| +-- -R-- Integer32 kpi53QNS02FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS02ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS02SessionCount(24)
| +-- -R-- Integer32 kpi53QNS02FreeMemory(25)

+--broadhopProductsQNSKPI53QNS03(43)
| |
| +-- -R-- Integer32 kpi53QNS03PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS03QueueSize(21)
| +-- -R-- Integer32 kpi53QNS03FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS03ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS03SessionCount(24)
| +-- -R-- Integer32 kpi53QNS03FreeMemory(25)

+--broadhopProductsQNSKPI53QNS04(44)
| |
```

| +-- -R-- Integer32 kpi53QNS04PolicyCount(20)

| +-- -R-- Integer32 kpi53QNS04QueueSize(21)

| +-- -R-- Integer32 kpi53QNS04FailedEnqueueCount(22)

| +-- -R-- Integer32 kpi53QNS04ErrorCount(23)

| +-- -R-- Integer32 kpi53QNS04SessionCount(24)

| +-- -R-- Integer32 kpi53QNS04FreeMemory(25)

Note Additional KPI's are provided in CPS 6.1 MIB for backward compatibility purpose but are no longer supported in this release.

Notifications and Alerting (Traps)

The CPS Monitoring and Alert Notification framework provides the following SNMPv2 notification traps (one-way). Traps are either proactive or reactive. Proactive traps are alerts based on system events or changes that require attention (for example, Disk is filling up). Reactive traps are alerts that an event has already occurred (e.g., an application process died).

This section covers the following topics:

- Component Notifications
- Application Notifications

Component Notifications

Components are devices that make up the CPS system. These are systems level traps. They are generated when some predefined thresholds are crossed. User can define these thresholds in `/etc/snmp/snmpd.conf`. For example, for disk full, low memory etc. Process snmpd is running on all the VMs. When process snmpd starts, it notes the values set in snmpd.conf. Hence, whenever user makes any change in snmpd.conf, the user must execute command

service snmpd restart

For example, if threshold crosses, snmpd throws a trap to LBVIP on the internal network on port 162. On LB, process snmptrapd is listening on port 162. When snmptrap sees trap on 162, it logs it in the file `/var/log/snmp/trap` and re-throws it on corporate_nms_ip on port 162. This corporate NMS IP is set inside `/etc/hosts` file on LB1 and LB2. Typically, these components equate to running Virtual Machines.

Note For more information related to `corportate_nms_ip` configuration, refer *SNMP Traps and Key Performance Indicators (KPIs)* section in *CPS 6.1 Troubleshooting Guide*.

Component notifications are defined in the BROADHOP-NOTIFICATION-MIB as follows:

broadhopQNSComponentNotification NOTIFICATION-TYPE

OBJECTS { broadhopComponentName,

broadhopComponentTime,

```
            broadhopComponentNotificationName,
            broadhopNotificationFacility,
            broadhopNotificationSeverity,
            broadhopComponentAdditionalInfo }
        STATUS current
        DESCRIPTION "
                Trap from any QNS component - i.e. device.
                "
        ::= { broadhopProductsQNSNotifications 1 }
```

Each Component Notification contains:

- Name of the device throwing the notification (broadhopComponentName)
- Time the notification was generated (broadhopComponentTime)
- Facility or which layer the notification came from (broadhopNotificationFacility)
- Severity of the error (broadhopNotificationSeverity)
- Additional information about the notification, which might be a bit of log or other information.

Component Notifications that CPS generates are shown in the following list. Any component in the CPS system may generate these notifications.

<table>
<tr><th>Name</th><th>Feature</th><th>Severity</th><th>Message Text</th></tr>
<tr><td rowspan="2">Disk Full: This alarm gets generated for following file system:
1. /
2. /var
3. /home
4. /boot
5. /opt</td><td>Component</td><td>Warning</td><td>DiskFullAlert</td></tr>
<tr><td colspan="3">Current disk usage has passed a designated threshold. This situation may resolve on its own, but could be a sign of logs or database files growing large.</td></tr>
<tr><td rowspan="2">Disk Full Clear: This alarm gets generated for following file system:
1. /
2. /var
3. /home
4. /boot
5. /opt</td><td>Component</td><td>Clear</td><td>DiskFullClear</td></tr>
<tr><td colspan="3">Current Disk usage has recovered the designated Threshold.</td></tr>
</table>

Name	Feature	Severity	Message Text
Load Average of local system: The alarm gets generated for 1 minute 5 minute 15 minute Average	Component	Warning (1, 5 minutes) Alert (15 minutes)	HighLoadAlert
	Current CPU load is more than configured threshold for 1/5/15 minutes.		
Load Average Clear of local system: The alarm gets generated for 1 minute 5 minute 15 minute Average	Component	Clear	HighLoadClear
	Current CPU load has recovered from more than configured threshold.		
Low Swap memory alarm	Operating System	Warning	LowSwapAlert
	Current swap usage has passed a designated threshold. This is a warning.		
Low Swap memory clear	Operating System	Clear	LowSwapClear
	Current swap usage has recovered a designated threshold.		
Interface Down Alarm: This alarm gets generated for all physical interface attached to the system.	Operating System	Alert	<Interface Name> is Down
	Not able to connect or ping to the interface.		
Interface Up Alarm: This alarm gets generated for all physical interface attached to the system.	Operating System	Clear	<Interface Name> is Up
	Able to ping or connect to interface.		
Low Memory Alert Alarm	Operating System	Warning	LowMemoryAlert
	Low memory alert.		
Low Memory Clear Alarm	Operating System	Info	LowMemoryClear
	Low memory alert.		

Application Notifications

Applications are running processes on a component device that make up the CPS system. These are application level traps. CPS process (starting with word java when we run "ps -ef") and some scripts (for GR traps) generates these traps.

For example, when a trap is generated, it is thrown to LBVIP on internal network (can be on port 162. On LB, process snmptrapd is listening on port 162. When snmptrap sees trap on 162, it logs it in the file `/var/log/snmpd/trap` and re-throws it on corporate_nms_ip on port 162. This corporate NMS IP is set inside /etc/hosts file on LB1 and LB2.

Note For more information related to `corportate_nms_ip` configuration, refer *SNMP Traps and Key Performance Indicators (KPIs)* section in *CPS 6.1 Troubleshooting Guide*.

Application notifications are defined in the BROADHOP-NOTIFICATION-MIB as follows:

```
broadhopQNSApplicationNotification NOTIFICATION-TYPE
    OBJECTS { broadhopComponentName,
          broadhopComponentTime,
          broadhopComponentNotificationName,
          broadhopNotificationFacility,
          broadhopNotificationSeverity,
          broadhopComponentAdditionalInfo }
    STATUS current
    DESCRIPTION "
            Notification Trap from any QNS application - i.e., runtime.
            "
    ::= { broadhopProductsQNSNotifications 2 }
```

Each Application Notification contains these elements:

- Name of the device throwing the notification (broadhopComponentName)
- Time the notification was generated (broadhopComponentTime)
- Facility or which layer the notification came from (broadhopNotificationFacility)
- Severity of the error (broadhopNotificationSeverity)
- Additional information about the notification, which might be a portion of log or other information

Application Notifications that CPS generates are shown in the following list. Any application in CPS system may generate these notifications.

Name	Feature	Severity	Message Text
License Usage Threshold Exceeded	Application	Major/Minor/Warning/Critical/Clear	"Session Count License Usage at: xx%,threshold at: xx%"
	The license threshold is defined and SNMP traps are sent out if the thresholds ere exceeded by the real license usage. This limits the total session count usage.		
Memcached ConnectError	Application	Critical	Memcached server is in error: %s
	Generated if the system is unable to connect to memcached server or unable to write to it.		
MemcachedConnectError	Application	Major	Memcached server is in error: OR some exception generated message.
	Generated if attempting to connect to or write to the memcached server causes an exception. %s is the exception that occurred.		
ApplicationStartError	Application	Alert	Feature %s is unable to start. Error %s
	Generated if an installed feature cannot start.		

Name	Feature	Severity	Message Text
LicensedSessionCreation	Application	Critical	Common Services: 4:Session creation is not allowed
	A predefined threshold of sessions covered by licensing has been passed. This is a warning and should be reported. License limits may need to be increased soon. This message can be generated by an invalid license, but the AdditionalInfo portion of the notification shows root cause.		
InvalidLicense	Application	Emergency	"xxx license has not been verified yet"
	The system license currently installed is not valid. This prevents system operation until resolved. This is possible if no license is installed or if the current license does not designate values. This also may occur if any system networking MAC addresses have changed..		
	Application	Emergency	xxx license is invalid %s
	License is invalid.		
	Application	Critical	xxx is Expired %s
	License has expired.		
	Application	Major	xxx will expire soon %s
	License is going to expire soon.		
	Application	Critical	xxx has exceeded the allowed parameters %s
	License has exceeded the allowed parameters.		
	Application	Major	xxx is nearing the allowed parameters %s
	License is nearing allowed parameters.		
PolicyConfiguration	Application	Major	"Last policy configuration failed with the following message:xxx"
	A change to system policy structure has failed. The AdditionalInfo portion of the notification contains more information. The system typically remains in a proper state and continues core operations. Either make note of this message or investigate more fully.		
SessionManagerUnavailable	Application	Major	"$HOST : $PORT session management node is not available"
	The reported session management node is unavailable. This should not immediately affect core services, but may degrade service performance.		
PoliciesNot	Application	Emergency	"Policies not configured "
Configured	The policy engine cannot find any policies to apply while starting up. This may occur on a new system, but requires immediate resolution for any system services to operate.		
DiameterPeerDown	Application	Major	host $HOST realm: %s is down
	Diameter peer is down.		

Name	Feature	Severity	Message Text
HA_Failover	Application	Critical	HA Failover done from %s to %s of ${SET_NAME}-SET$Loop
	Primary member of replica is down and taken over by other primary member of same replica set.		
Geo_Failover	Application	Critical	Geo Failover done from %s to %s of ${SET_NAME}-SET$Loop
	Primary member of replica set is down and taken over by other member of the same replica set.		
All_replica_of_DB_ down	Application	Critical	All replicas of ${SET_NAME}-SET$Loop are down
	Not able to connect to any member of replica set.		
No_Primary_membe r_found	Application	Critical	Unable to find primary member for Replica-set ${SET_NAME}-SET$Loop
	Unable to find primary member for replica-set.		
Secondary DB Down	Application	Critical	Secondary DB %member_ip:%mem_port (%mem_hostname) of SET $SET is down
	In replica set, secondary DB member is not able to connect.		
Arbiter Down	Application	Critical	Arbiter %member_ip:%mem_port (%mem_hostname) of SET $SET is down
	In replica set, the administrator is not able to connect to configured arbiter.		
Config Server is Down	Application	Critical	Config Server %member_ip:%mem_port (%mem_hostname) of SET $SET is down
	In replica set, the administrator is not able to connect to Configured Config Server.		
Site Down	Application	Critical	Site %site is down
	The administrator is not able to connect to the Site.		
VM Down	Application	Critical	unable to connect %member_ip (%member) VM. It is not reachable.
	The administrator is not able to ping to VM (This alarms gets generated for all VMs configured inside /etc/hosts of lb).		
QPS Process Down	Application	Critical	%server server on %vm vm is down
	QPS java process is down.		
Admin Logged In	Application	Critical	root user logged in on %hostname terminal %terminal from machine %from_system
	root user logged in on %hostname terminal.		

Name	Feature	Severity	Message Text
VIRTUALInterfaceError	Network	Alert	unable to connect %INTERFACE(lbvip01/lbvip02) VM. Not reachable
	Not able to ping the virtual Interface. This alarm gets generated for lbvip01, lbvip02.		
Developer Mode License traps	Application	Warning	Common Services: 4:Using POC/Development license (100 session limit). To use a license file, remove -Dcom.broadhop.developer.mode from /etc/broadhop/qns.conf
	Generated if developer mode is configured in qns.conf.		
Can't create sessions due to errors	Application	Major	Session creation is not allowed
	Generated if license is not installed or session count has reached allowed license level.		
ZeroMQConnection Error	Application	Error	QNS Failed to Send Message:tcp://%s:%d
	Internal services cannot connect to a required Java ZeroMQ queue. Although retry logic and recovery is available, and core system functions should continue, investigate and remedy the root cause.		
ZeroMQConnection Error(clear)	Application	Clear	Send Message Success:tcp://%s:%d
	Internal services can connect to a required Java ZeroMQ queue.		

Configuration and Usage

All access to system statistics and KPIs should be collected via SNMP gets and walks from the virtual IP lbvip01, which can be located on either lb01 or lb02 load balancers. System notifications are also sourced from this address.

Configuration of the system consists of following:

- Configuration for SNMP gets and walks
- Configuration for Notifications (traps)
- License Usage Threshold
- Validation and Testing

Configuration for SNMP gets and walks

At the time of installation, SNMPv2 gets and walks can be performed against the system lbvip01 with the default read-only community string of Cisco using standard UDP port 161. The IP address of lbvip01 can be found in the `/etc/hosts` file of pcrfclient01, lb01 or lb02.

The read-only community string can be changed from its default of Cisco to a new value using the following steps:

Step 1 ssh to lb01 as the root user.

Step 2 With vi or nano, edit the file `/etc/snmp/snmpd.conf`.

Step 3 DO NOT CHANGE THE EXISTING line that reads: `rocommunity Broadhop`. Changing this line breaks SNMP framework functionality

Step 4 Add a NEW rocommunity line under the exiting with your new read-only string. The line should read: `rocommunity <string>`. Replace `<string>` with your desired community string value.

Step 5 Save and exit the file editor.

Step 6 scp `/etc/snmp/snmpd.conf lb02:/etc/snmp`.

Step 7 service snmpd restart

Step 8 ssh lb02 "service snmpd restart"

SNMPv2 gets and walks should now be accessible via the new rocommunity string. Changing the port from the default of 161 is not covered by this guide.

Caution Please do not change existing values of rocommunity, or trap2sink - this prevents the SNMP framework from functioning correctly.

Configuration for Notifications (traps)

After the previous configurations have been made, notifications should be logged locally in the `/var/log/snmp/trap` file as well as forwarded to the NMS destination at corporate_nms_ip. By default, traps are sent to the destination corporate_nms_ip using the SNMPv2 community string of Cisco. The standard SNMP UDP trap port of 162 is also used. Both of these values may be changed to accommodate the upstream NMS.

To change the trap community string:

Step 1 Ensure the SNMP framework has been installed.

Step 2 ssh to lb01 as the root user.

Step 3 With vi or nano, edit the file /etc/snmp/scripts/snmp_communities

Step 4 Edit the existing line that reads: `trap_community=broadhop`.

The changed line should read: `trap_community=<string>`.

Replace `<string>` with your desired trap community value.

Step 5 Save and exit the file editor.

Step 6 scp /etc/snmp/scripts/snmp_communities lb02:/etc/snmp/scripts

Step 7 service snmpd restart

Step 8 ssh lb02 "service snmpd restart"

To change the destination trap port from 162:

Step 1 To make this change, the `/etc/snmp/snmptrapd.conf` file needs modification on both lb01 and lb02. In these files,

a. Append a colon and the destination port to each line containing corporate_nms_ip. There are a total of 12 lines in each file.

For example, if the NMS destination port were 1162, the line:

```
traphandle DISMAN-EVENT-MIB::mteTriggerFired
/etc/snmp/scripts/component_trap_convert corporate_nms_ip
```

becomes

```
traphandle DISMAN-EVENT-MIB::mteTriggerFired
/etc/snmp/scripts/component_trap_convert corporate_nms_ip:1162
```

Step 2 After these changes, save the file and restart the snmptrapd service to enable changes.

License Usage Threshold

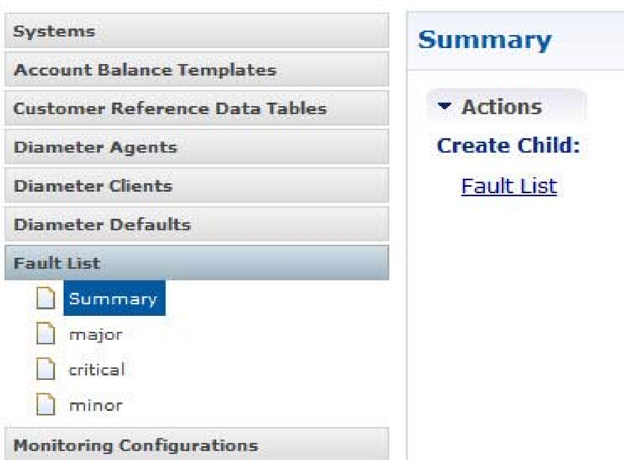
The license threshold can be defined and SNMP traps are sent out if the thresholds were exceeded by the real license usage. This limits the total session count usage.

This section covers the following topics:

- Policy Builder Configuration

Policy Builder Configuration

Step 1 In Policy Builder, select Reference Data > Fault List, click Fault List.



Step 2 Click Fault List to open a pane on the right side.

Step 3 Create a license usage threshold by:

a. Entering a name in Name field.

b. Enter License Usage Threshold Percentage.

c. Select alarm severity from Alarm Severity drop-down list (Critical/Major/Minor/Warning).

Note If two faults are created with the same threshold but different severity level, only one is applied but there is no guarantee on which severity will be in the trap.

If the condition of the license limit alarm goes away, then the auto-clear is sent and the license usage threshold alarm is cleared.

Validation and Testing

This section describes the commands for validation and testing of the CPS SNMP infrastructure during its development. You can use these commands now to validate and test your system during setup, configuration, or at any point. Our examples use MIB values because they are more descriptive, but you may use equivalent OID values if you like, particularly when configuring an NMS.

The examples here use Net-SNMP snmpget, snmpwalk and snmptrap programs. Detailed configuration of this application is outside the scope of this document, but the examples assume that the three Cisco MIBs are installed in the locations described on the man page of snmpcmd (typically the /home/share/<user>/.snmp/mibs or /usr/share/snmp/mibs directories).

Validation and testing is of three types and correspond to the statistics and notifications detailed earlier in this document:

- Component Statistics
- Application KPI
- Notifications

Run all tests from a client with network access to the Management Network or from the lb01, lb02, pcrfclient01 or pcrfclient02 hosts (which are also on the Management Network).

Component Statistics

Component statistics can be obtained on a per statistic basis with snmpget. As an example, to get the current available memory on pcrfclient01 use the following command:

snmpget -v 2c -c broadhop -m +BROADHOP-MIB:BROADHOP-QNS-MIB <lbvip01> BROADHOP-QNS-MIB::component53PCRFClient01MemoryAvailable

where `<lbvip01>` is the IP address of lbvip01 or as resolved from the `/etc/hosts` file. An example of the output from this command is:

BROADHOP-QNS-MIB::component53PCRFClient01MemoryAvailable = INTEGER: 629100

Interpret this output means that 629,100 MB of memory are available on this component machine.

All available component statistics in an MIB node can be “walked” via the snmpwalk command. This is very similar to snmpget as above. For example, to see all statistics on lb01 use the command:

snmpwalk -v 2c -c broadhop -m +BROADHOP-MIB:BROADHOP-QNS-MIB <lbvip01> BROADHOP-QNS-MIB::broadhopProductsQNSComponents53LB01

where `<lbvip01>` is the IP address of lbvip01 or as resolved from the /etc/hosts file. An example of the output from this command is:

BROADHOP-QNS-MIB::component53LB01CpuUser = INTEGER: 0

BROADHOP-QNS-MIB::component53LB01CpuSystem = INTEGER: 0

BROADHOP-QNS-MIB::component53LB01CpuIdle = INTEGER: 98

BROADHOP-QNS-MIB::component53LB01LoadAverage1 = INTEGER: 1

BROADHOP-QNS-MIB::component53LB01LoadAverage5 = INTEGER: 0

BROADHOP-QNS-MIB::component53LB01LoadAverage15 = INTEGER: 0

BROADHOP-QNS-MIB::component53LB01MemoryTotal = INTEGER: 1927736

BROADHOP-QNS-MIB::component53LB01MemoryAvailable = INTEGER: 590772

BROADHOP-QNS-MIB::component53LB01SwapTotal = INTEGER: 1048568

BROADHOP-QNS-MIB::component53LB01SwapAvailable = INTEGER: 1048568

BROADHOP-QNS-MIB::component53LB01Eth0InOctets = Counter32: 2047724337

BROADHOP-QNS-MIB::component53LB01Eth0OutOctets = Counter32: 3307198774

BROADHOP-QNS-MIB::component53LB01Eth1InOctets = Counter32: 2621101867

BROADHOP-QNS-MIB::component53LB01Eth1OutOctets = Counter32: 179361352

Application KPI

Application KPI can be obtained on a per statistic basis with snmpget in a manner much like obtaining Component Statistics. As an example, to get the number of sessions currently active on qns01, use the following command:

```
snmpget -v 2c -c broadhop -m +BROADHOP-MIB:BROADHOP-QNS-MIB <lbvip01>
    BROADHOP-QNS-MIB::kpi53QNS01SessionCount
```

where `<lbvip01>` is the IP address of lbvip01 or as resolved from the /etc/hosts file. An example of the output from this command would be:

```
BROADHOP-QNS-MIB::kpi53QNS01SessionCount = STRING: 937
```

Read this output means that 937 sessions are active on qns01.

Similarly, all available KPI in an MIB node can be “walked” via the snmpwalk command. This is very similar to snmpget as above. As an example, to see all statistics on qns02, use the following command:

```
snmpwalk -v 2c -c broadhop -m +BROADHOP-MIB:BROADHOP-QNS-MIB <lbvip01>
    BROADHOP-QNS-MIB::broadhopProductsQNSKPI53QNS02
```

where `<lbvip01>` is the IP address of lbvip01 or as resolved from the /etc/hosts file. An example of the output from this command would be:

```
BROADHOP-QNS-MIB::kpi53QNS02PolicyCount = STRING: 4

BROADHOP-QNS-MIB::kpi53QNS02QueueSize = STRING: 0

BROADHOP-QNS-MIB::kpi53QNS02FailedEnqueueCount = STRING: 0

BROADHOP-QNS-MIB::kpi53QNS02ErrorCount = STRING: 0

BROADHOP-QNS-MIB::kpi53QNS02SessionCount = STRING: 937

BROADHOP-QNS-MIB::kpi53QNS02FreeMemory = STRING: 3721598032
```

Notifications

Testing and validating notifications requires slightly more skill than testing SNMP gets and walks. Recall that the overall architecture is that all components and applications in the CPS system are configured to send notifications to lb01 or lb02 via lbvip02, the Internal Network IP. These systems log the notification locally in `/var/log/snmp/trap` and then “re-throw” the notification to the destination configured by corporate_nms_ip. Two testing and troubleshooting methods are illustrated below: confirming notifications are being sent properly from system components to lb01 or lb02, and confirming that notifications can be sent upstream to the NMS.

Receiving Notifications

There are several ways to confirm that lb01 or lb02 are properly receiving notifications from components. First, determine the active load balancer – it is either lb01 or lb02 and have multiple IP addresses per interface as shown by the `ifconfig` command.

Step 1 Log in to the active load balancer with ssh as the root user.

Note Use the ifconfig command to identify the active load balancer, either lbvip01 or lbvip02.

Step 2 Look at the trap log on the active load balancer in "follow" mode with the command tail `-f /var/log/snmp/trap`.

Step 3 Note the last trap thrown.

Step 4 In a separate window log into another "safe" system – qns04 as the root user.

Note Safe system should not be a production system or a system processing live customer traffic.

Step 5 A LowSwap warning notification can be thrown from qns04 to the active load balancer by temporarily turning off swap. This is relatively safe, but may have production impacts so care is needed. Turn off swap with the command:

```
swapoff -a
```

Step 6 Wait up to 2 minutes for the snmp daemon to detect and throw a trap.

Step 7 Note the notification arrive in the active load balancer in the "tail"-ed log. This is seen in the logs as a detailed trap message and a logger message that reads similar to: Mar 20 17:20:00 lb01 logger: Forwarded qns04 LowSwap to `<corporate_nms_ip>`.

Step 8 Turn swap back on in qns04 with the command:

```
swapon -a
```

If a LowSwap notification is received at the active load balancer, then this notification is re-thrown to the destination NMS. Check the NMS logs for this notification.

Upstream Notifications

Should a notification not be received by the NMS, you can manually throw a notification from the active load balancer to the NMS using this command:

```
snmptrap -v 2c -c broadhop <corporate_nms_ip> ""
    NET-SNMP-EXAMPLES-MIB::netSnmpExampleHeartbeatNotification
    netSnmpExampleHeartbeatRate i 123456
```

where `<corporate_nms_ip>` is the appropriate NMS IP address. This sends an SNMPv2 trap from the active load balancer to the NMS and can be used for debugging.

Reference Document

For more information related to SNMP Traps and KPIs, refer to *CPS 6.1 Troubleshooting Guide*.

Data Collection and KPI

Revised: August 28, 2014,

This chapter covers the following sections:

- Data Collection
- Key Performance Indicators (KPIs)
- Example Data

Data Collection

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 centralized collection nodes on the pcrfclient01 machine. The centralized collector can then write the data to a number of available output formats and destinations. These output formats include:

Output Type	Destination	Reference
CSV	Local	https://collectd.org/wiki/index.php/Plugin:CSV
RRD	Local	https://collectd.org/wiki/index.php/Plugin:RRDtool
RRD Caching Daemon	Remote	https://collectd.org/wiki/index.php/Plugin:RRDCacheD
Graphite	Remote	https://collectd.org/wiki/index.php/Plugin:Write_Graphite
HTTP Json	Remote	https://collectd.org/wiki/index.php/Plugin:Write_HTTP
HTTP Text	Remote	https://collectd.org/wiki/index.php/Plugin:Write_HTTP

The collector by defaults is configured to collect the following data from the nodes in the system:

Data Collected	Plugin	Reference
Free Page Cache Buffer Cache Used	memory	https://collectd.org/wiki/index.php/Plugin:Memory
Per Virtual Core: • Idle • Interrupt • Nice • User • System • Wait • SoftIRQ	cpu	https://collectd.org/wiki/index.php/Plugin:CPU
Merged Ops Ops Time Octets	disk	https://collectd.org/wiki/index.php/Plugin:Disk
Disk Space	df	https://collectd.org/wiki/index.php/Plugin:DF
Octets Errors Packets	Interface	https://collectd.org/wiki/index.php/Plugin:Interface
Application TPS Rates	JMX	Refer to CSV Formatted Data

CSV Formatted Data

For CSV formatted data, the collector writes to the following directories on the pcrfclient01 and 02 on a 10 second interval:

/var/lib/collectd/csv/{host name}/{metric-group}/{metric}-YYYY-MM-DD

Host Name	Metric Group	Metric	CSV Format
Linux Host Name	cpu-x	cpu-idle cpu-interrupt cpu-nice cpu-softirq cpu-steal cpu-system cpu-user cpu-wait	Timestamp (Seconds with 3 decimal precision), Linux "jiffies" (rate)
	df-home	df_complex-used df_complex-reserved df_complex-free	Timestamp (Seconds with 3 decimal precision), Bytes (rate)
	df-root	df_complex-used df_complex-reserved df_complex-free	Timestamp (Seconds with 3 decimal precision), Bytes (rate)
	df-var	df_complex-used df_complex-reserved df_complex-free	Timestamp (Seconds with 3 decimal precision), Bytes (rate)
	disk-sdax	disk_merged disk_octets disk_ops disk_time	Timestamp (Seconds with 3 decimal precision), Read Value, Write Value Value is in the following units (rates): • Merged Operations • Octets • Operations • Time (seconds)
	exec	server_availability	Timestamp (Seconds with 3 decimal precision), 1 or 0 indicating availability
	exec	qns_availability	Timestamp (Seconds with 3 decimal precision), 1 or 0 indicating availability
	interface-ethx	if_packets if_octets if_errors	Timestamp (Seconds with 3 decimal precision), Read Value (rate), Write Value (rate)

Host Name	Metric Group	Metric	CSV Format
	memory	memory-buffered memory-cached memory-free memory-used	Timestamp (Seconds with 3 decimal precision), Bytes (gauge)
	GenericJMX-{tpsmetric}	qns_count	Timestamp (Seconds with 3 decimal precision), Count (rate)

Note "x" represents the different metric group number.

The system supports generation of TPS metrics on incoming and outgoing diameter messages. The (tpsmetric) is generated at system runtime when the application encounters a message. If a metric file does not exist then it is assumed that the value is "0". The tpsmetric is generated using the following two conventions:

{diameter realm}_{protocol}

{diameter realm}_{protocol}_{action | response code}

Part	Description/Valid Values
{diameter realm}	Realm on the diameter message if known.
{protocol}	Gx_RAR, Gx_RAA, Gx_CCR-I, Gx_CCA-I, Gx_CCR-U, Gx_CCA-U, Gx_CCR-T, Gx_CCA-T As more protocol are enabled, more values will be added.
{action | response code}	bypass, rate-limit, rate-limit-outbound sync, drop, send-x, pe-submit-error, pe-prc-error, {diameter result code}

Key Performance Indicators (KPIs)

This sections covers the following topics:

- Gx KPI
- Sy KPI
- Sh KPI

Gx KPI

The KPIs are available in the CSV files documented in the section CSV Formatted Data, and all the KPIs in this section are prefixed with realm name configured for the Gx interface in the policy builder.

KPI NAme	Description
Gx_RAR	Counter, number of Re-Auth-Request Messages processed.
Gx_RAA	Counter, number of Re-Auth-Answer Messages processed.
Gx_RAA_drop	Counter, number of Re-Auth-Answer Messages dropped due to internal overload protection
Gx_RAA_bypass	Counter, number of Re-Auth-Answer Messages bypassed the rules engine because of successful responses.
Gx_RAA_Rate-limit	Counter, number of Re-Auth-Answer Messages dropped due to the configured rate limit control in the policy builder.
Gx_RAA_Rate-limit-outbound	Counter, number of Re-Auth-Answer Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Gx_RAA_sync	Counter, number of Re-Auth-Answer Messages processed as synchronized response messages.
Gx_RAA_Send-x	Counter, number of Re-Auth-Answer Messages failed to send out.
Gx_RAA_Pe-submit-error	Counter, number of Re-Auth-Answer Messages failed to be submitted to the policy engine.
Gx_RAA_Pe-prc-error	Counter, number of Re-Auth-Answer Messages failed to process in the outbound queue.
Gx_CCR-I	Counter, number of Initial Credit Control Request Messages processed.
Gx_CCR-I_drop	Counter, number of Initial Credit Control Request Messages dropped due to internal overload protection.
Gx_CCR-I_bypass	Counter, number of Initial Credit Control Request Messages bypassed the rules engine because of successful responses.
Gx_CCR-I_Rate-limit	Counter, number of Initial Credit Control Request Messages dropped due to the configured rate limit control in the policy builder.
Gx_CCR-I_Rate-limit-outbound	Counter, number of Initial Credit Control Request Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Gx_CCR-I_sync	Counter, number of Initial Credit Control Request Messages processed as synchronized response messages.

KPI NAme	Description
Gx_CCR-I_Send-x	Counter, number of Initial Credit Control Request Messages failed to send out.
Gx_CCR-I_Pe-submit-error	Counter, number of Initial Credit Control Request Messages failed to be submitted to the policy engine.
Gx_CCR-I_Pe-prc-error	Counter, number of Initial Credit Control Request Messages failed to process in the outbound queue.
Gx_CCA-I	Counter, number of Credit Control Answer Messages.
Gx_CCR-T	Counter, number of Terminate Credit Control Request Messages.
Gx_CCR-T_drop	Counter, number of Terminate Credit Control Request Messages dropped due to internal overload protection.
Gx_CCR-T_bypass	Counter, number of Terminate Credit Control Request Messages bypassed the rules engine because of successful responses.
Gx_CCR-T_Rate-limit	Counter, number of Terminate Credit Control Request dropped due to the configured rate limit control in the policy builder Messages.
Gx_CCR-T_Rate-limit-outbound	Counter, number of Terminate Credit Control Request Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Gx_CCR-T_sync	Counter, number of Terminate Credit Control Request Messages processed as synchronized response messages.
Gx_CCR-T_Send-x	Counter, number of Terminate Credit Control Request Messages failed to send out.
Gx_CCR-T_Pe-submit-error	Counter, number of Terminate Credit Control Request Messages failed to be submitted to the policy engine.
Gx_CCR-T_Pe-prc-error	Counter, number of Terminate Credit Control Request Messages failed to process in the outbound queue.
Gx_CCA-T	Counter, number of Terminate Credit Control Answer Messages processed.
Gx_CCR-U	Counter, number of Credit Control Request Update Messages processed.
Gx_CCR-U_drop	Counter, number of Credit Control Request Update Messages dropped due to internal overload protection.
Gx_CCR-U_bypass	Counter, number of Credit Control Request Update Messages bypassed the rules engine because of successful responses.
Gx_CCR-U_Rate-limit	Counter, number of Credit Control Request Update Messages dropped due to the configured rate limit control in the policy builder.
Gx_CCR-U_Rate-limit-outbound	Counter, number of Credit Control Request Update Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Gx_CCR-U_sync	Counter, number of Credit Control Request Update Messages processed as synchronized response messages.
Gx_CCR-U_Send-x	Counter, number of Credit Control Request Update Messages failed to send out.
Gx_CCR-U_Pe-submit-error	Counter, number of Credit Control Request Update Messages failed to be submitted to the policy engine.

KPI NAme	Description
Gx_CCR-U_Pe-prc-error	Counter, number of Credit Control Request Update Messages failed to process in the outbound queue.
Gx_CCA-U	Counter, number of Credit Control Request Update Messages processed.

Sy KPI

The KPIs are available in the CSV files documented in the section CSV Formatted Data, and all the KPIs in this section are prefixed with realm name configured for the Sy interface in the policy builder.

KPI NAme	Description
Sy_STR	Counter, number of Session-Termination-Request Messages processed.
Sy_STA	Counter, number of Session-Termination-Answer Messages processed.
Sy_STA_drop	Counter, number of Session-Termination-Answer Messages dropped due to internal overload protection.
Sy_STA_bypass	Counter, number of Session-Termination-Answer Messages bypassed the rules engine because of successful responses.
Sy_STA_Rate-limit	Counter, number of Session-Termination-Answer Messages dropped due to the configured rate limit control in the policy builder.
Sy_STA_Rate-limit-outbound	Counter, number of Session-Termination-Answer Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sy_STA_sync	Counter, number of Session-Termination-Answer Messages processed as synchronized response messages.
Sy_STA_Send-x	Counter, number of Session-Termination-Answer Messages failed to send out.
Sy_STA_Pe-submit-error	Counter, number of Session-Termination-Answer Messages failed to be submitted to the policy engine.
Sy_STA_Pe-prc-error	Counter, number of Session-Termination-Answer Messages failed to process in the outbound queue.
Sy_SLR	Counter, number of Spending-Limit-Request Messages processed.
Sy_SLA	Counter, number of Spending-Limit-Answer Messages processed.
Sy_SLA_drop	Counter, number of Spending-Limit-Answer Messages dropped due to internal overload protection.
Sy_SLA_bypass	Counter, number of Spending-Limit-Answer Messages bypassed the rules engine because of successful responses.
Sy_SLA_Rate-limit	Counter, number of Spending-Limit-Answer Messages dropped due to the configured rate limit control in the policy builder.
Sy_SLA_Rate-limit-outbound	Counter, number of Spending-Limit-Answer Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sy_SLA_sync	Counter, number of Spending-Limit-Answer Messages processed as synchronized response messages.
Sy_SLA_Send-x	Counter, number of Spending-Limit-Answer Messages failed to send out.

KPI NAme	Description
Sy_SLA_Pe-submit-error	Counter, number of Spending-Limit-Answer Messages failed to be submitted to the policy engine.
Sy_SLA_Pe-prc-error	Counter, number of Spending-Limit-Answer Messages failed to process in the outbound queue.
Sy_SSNR	Counter, number of Spending-Status-Notification-Request Messages processed.
Sy_SSNR drop	Counter, number of Spending-Status-Notification-Request Messages dropped due to internal overload protection.
Sy_SSNR bypass	Counter, number of Spending-Status-Notification-Request Messages bypassed the rules engine because of successful responses.
Sy_SSNR Rate-limit	Counter, number of Spending-Status-Notification-Request Messages dropped due to the configured rate limit control in the policy builder.
Sy_SSNR Rate-limit-outbound	Counter, number of Spending-Status-Notification-Request Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sy_SSNR sync	Counter, number of Spending-Status-Notification-Request Messages processed as synchronized response messages.
Sy_SSNR Send-x	Counter, number of Spending-Status-Notification-Request Messages failed to send out.
Sy_SSNR Pe-submit-error	Counter, number of Spending-Status-Notification-Request Messages failed to be submitted to the policy engine.
Sy_SSNR Pe-prc-error	Counter, number of Spending-Status-Notification-Request Messages failed to process in the outbound queue.
Sy_SSNR	Counter, number of Spending-Status-Notification-Request Messages processed.
Sy_SSNA	Counter, number of Messages.

Sh KPI

The KPIs are available in the CSV files documented in the section CSV Formatted Data, and all the KPIs in this section are prefixed with realm name configured for the Sh interface in the policy builder.

KPI NAme	Description
Sh_UDR	Counter, number of User-Data-Request Messages processed.
Sh_UDA	Counter, number of User-Data-Answer Messages processed.
Sh_UDA_drop	Counter, number of User-Data-Answer Messages dropped due to internal overload protection.
Sh_UDA_bypass	Counter, number of User-Data-Answer Messages bypassed the rules engine because of successful responses.
Sh_UDA_Rate-limit	Counter, number of User-Data-Answer Messages dropped due to the configured rate limit control in the policy builder.

KPI NAme	Description
Sh_UDA_Rate-limit-outbound	Counter, number of User-Data-Answer Messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sh_UDA_sync	Counter, number of User-Data-Answer messages processed as synchronized response messages.
Sh_UDA_Send-x	Counter, number of User-Data-Answer Messages failed to send out.
Sh_UDA_Pe-submit-error	Counter, number of User-Data-Answer messages failed to be submitted to the policy engine.
Sh_UDA_Pe-prc-error	Counter, number of User-Data-Answer messages failed to process in the outbound queue.
Sh_PUR	Counter, number of Profile-Update-Request messages processed.
Sh_PUA	Counter, number of Profile-Update-Answer messages.
Sh_PUA_drop	Counter, number of Profile-Update-Answer messages dropped due to internal overload protection.
Sh_PUA_bypass	Counter, number of Profile-Update-Answer messages bypassed the rules engine because of successful responses.
Sh_PUA_Rate-limit	Counter, number of Profile-Update-Answer messages dropped due to the configured rate limit control in the policy builder.
Sh_PUA_Rate-limit-outbound	Counter, number of Profile-Update-Answer messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sh_PUA_sync	Counter, number of Profile-Update-Answer messages processed as synchronized response messages.
Sh_PUA_Send-x	Counter, number of Profile-Update-Answer messages failed to send out.
Sh_PUA_Pe-submit-error	Counter, number of Profile-Update-Answer messages failed to be submitted to the policy engine.
Sh_PUA_Pe-prc-error	Counter, number of Profile-Update-Answer messages failed to process in the outbound queue.
Sh_SNR	Counter, number of Subscribe-Notifications-Request messages.
Sh_SNA	Counter, number of Subscribe-Notifications-Answer messages.
Sh_SNA_drop	Counter, number of Subscribe-Notifications-Answer messages dropped due to internal overload protection.
Sh_SNA_bypass	Counter, number of Subscribe-Notifications-Answer messages bypassed the rules engine because of successful responses.
Sh_SNA_Rate-limit	Counter, number of Subscribe-Notifications-Answer messages dropped due to the configured rate limit control in the policy builder.
Sh_SNA_Rate-limit-outbound	Counter, number of Subscribe-Notifications-Answer messages dropped due to the configured rate limit control in the policy builder for outbound messages.
Sh_SNA_sync	Counter, number of Subscribe-Notifications-Answer messages processed as synchronized response messages.
Sh_SNA_Send-x	Counter, number of Subscribe-Notifications-Answer messages failed to send out.

KPI NAme	Description
Sh_SNA_Pe-submit-error	Counter, number of Subscribe-Notifications-Answer messages failed to be submitted to the policy engine.
Sh_SNA_Pe-prc-error	Counter, number of Subscribe-Notifications-Answer messages failed to process in the outbound queue.
SH_PNR	Counter, number of Push-Notification-Request messages.
SH_PNR_drop	Counter, number of Push-Notification-Request messages dropped due to internal overload protection.
SH_PNR_bypass	Counter, number of Push-Notification-Request messages bypassed the rules engine because of successful responses.
SH_PNR_Rate-limit	Counter, number of Push-Notification-Request messages dropped due to the configured rate limit control in the policy builder.
SH_PNR_Rate-limit-outbound	Counter, number of Push-Notification-Request messages dropped due to the configured rate limit control in the policy builder for outbound messages.
SH_PNR_sync	Counter, number of Push-Notification-Request messages processed as synchronized response messages.
SH_PNR_Send-x	Counter, number of Push-Notification-Request messages failed to send out.
SH_PNR_Pe-submit-error	Counter, number of Push-Notification-Request messages failed to be submitted to the policy engine.
SH_PNR_Pe-prc-error	Counter, number of Push-Notification-Request messages failed to process in the outbound queue.
SH_PNA	Counter, number of Push-Notification-Answer messages processed.

Example Data

qps@qps-WS01 cd QPS_sample_stats/csv/qns10

$ ls -l

total 64

d---------+ 1 qps None 0 Feb 21 07:03 cpu-0

d---------+ 1 qps None 0 Feb 21 07:03 cpu-1

d---------+ 1 qps None 0 Feb 21 07:03 cpu-2

d---------+ 1 qps None 0 Feb 21 07:03 cpu-3

d---------+ 1 qps None 0 Feb 21 07:03 cpu-4

d---------+ 1 qps None 0 Feb 21 07:03 cpu-5

d---------+ 1 qps None 0 Feb 21 07:03 df-home

d---------+ 1 qps None 0 Feb 21 07:03 df-root

```
d---------+ 1 qps None 0 Feb 21 07:03 df-var
d---------+ 1 qps None 0 Feb 21 07:03 disk-sda
d---------+ 1 qps None 0 Feb 21 07:03 disk-sda1
d---------+ 1 qps None 0 Feb 21 07:03 disk-sda2
d---------+ 1 qps None 0 Feb 21 07:03 exec
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-cisco.com_Gx_RAR
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-cisco.com_msg_Gx_RAR
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-ciscos.com_msg_Sy_STR
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-ciscos.com_Sy_STR
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-null_msg_Sy_STA
d---------+ 1 qps None 0 Feb 21 07:03 GenericJMX-null_Sy_STA_3004
d---------+ 1 qps None 0 Feb 21 07:03 interface-eth0
d---------+ 1 qps None 0 Feb 21 07:03 interface-lo
d---------+ 1 qps None 0 Feb 21 07:03 interface-sit0
d---------+ 1 qps None 0 Feb 21 07:03 load
d---------+ 1 qps None 0 Feb 21 07:03 memory

qns@qns-WS01 QPS_sample_stats/csv/qns10
$ cd GenericJMX-cisco.com_Gx_RAR/

qns@qns-WS01 ../qns10/GenericJMX-cisco.com_Gx_RAR
$ head qns_count-2014-02-20
epoch,Count
1392952790.319,0.000000
1392952800.319,0.000000
1392952810.317,0.000000
1392952820.319,0.000000
1392952830.319,0.000000
1392952840.319,0.000000
```

1392952850.320,0.000000

1392952860.319,0.000000

1392952870.319,0.000000