Cisco Policy Suite 6.0 Alarming and SNMP Guide

Version 6.0

December 17, 2013

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.

Text Part Number: OL-30943-01

Preface

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.

This preface covers the following topics:

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

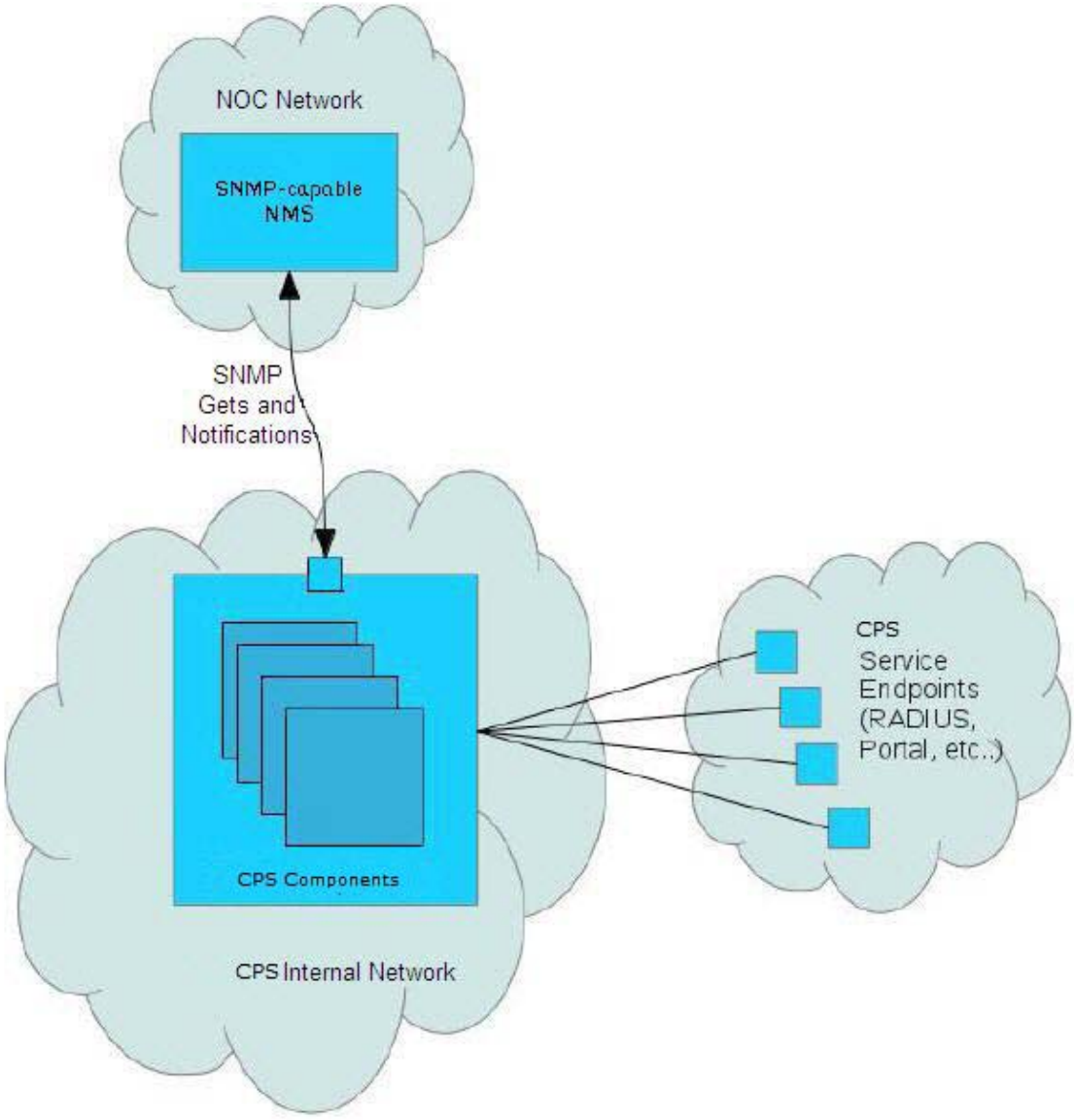
Revised: February 14, 2014, OL-30943-01

This chapter covers the following sections:

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 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.

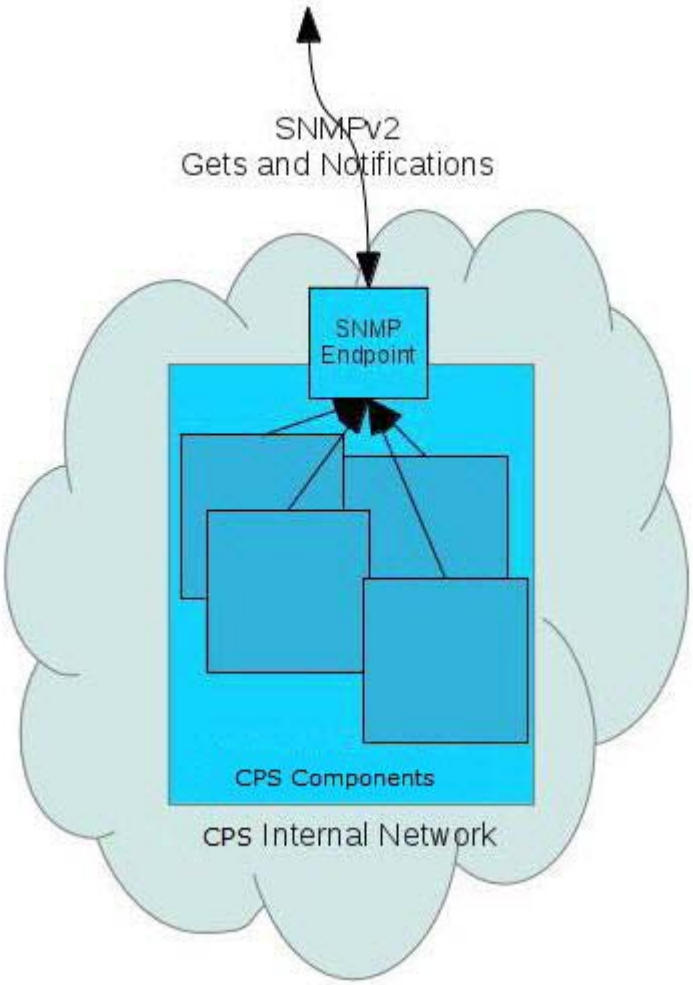
This section covers the following topics:

- 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.



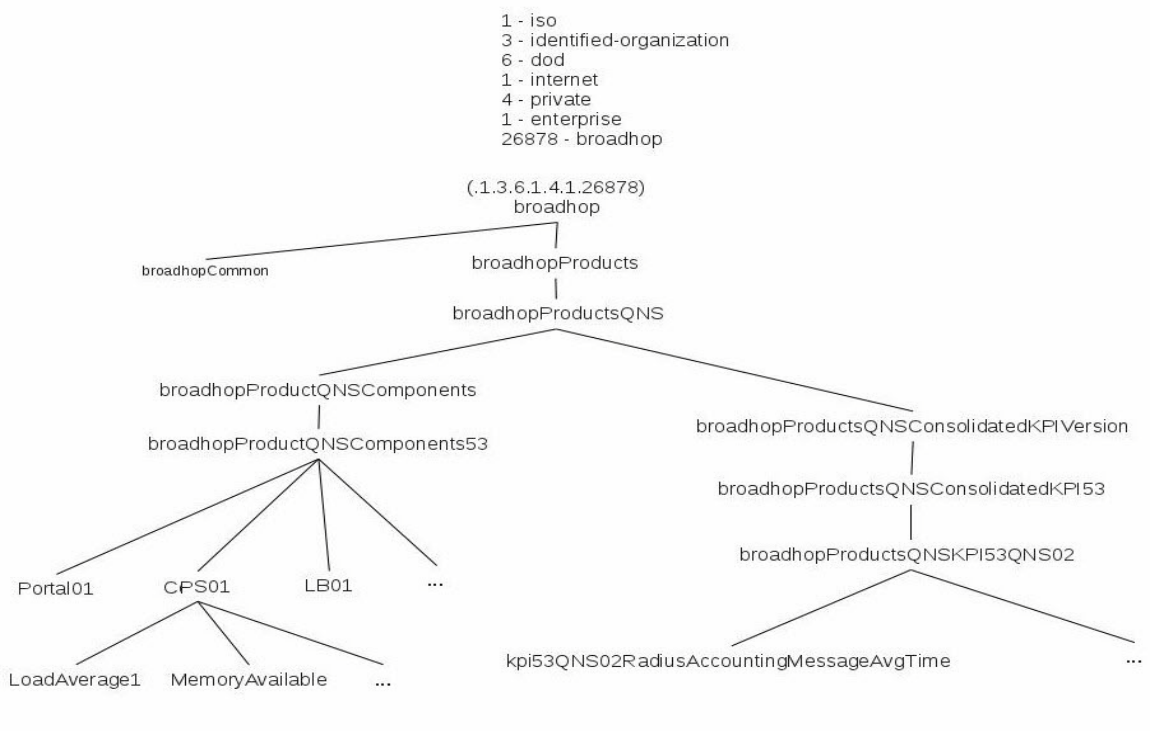
SNMP Object Identifier and Management Information Base

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 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 processes, 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, Cisco sessionmgr, etc.
5	Process	Particular httpd process, CPS 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.

Monitoring, Trending, and Key Performance Indicators

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:

Summary of Monitoring and Trending

In this table, the components listed on the left provide the information listed on the right. That is, for monitoring and trending, all components provide the same information.

Component	Information
LB01	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets
LB02	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Component	Information
PortalLB01	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets
PortalLB02	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Component	Information
PCRFClient01	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets
PCRFClient02	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Component	Information
SessionMgr01	CpuUser CpuSystem CpuIdle LoadAverage1 LoadAverage5 LoadAverage15 MemoryTotal MemoryAvailable SwapTotal SwapAvailable Eth0InOctets Eth0OutOctets Eth1InOctets Eth1OutOctets
SessionMgr02	CpuUser CpuSystem CpuIdle LoadAverage1 LoadAverage5 LoadAverage15 MemoryTotal MemoryAvailable SwapTotal SwapAvailable Eth0InOctets Eth0OutOctets Eth1InOctets Eth1OutOctets

Component	Information
QNS01	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets
QNS02	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Component	Information
QNS03	CpuUser CpuSystem CpuIdle LoadAverage1 LoadAverage5 LoadAverage15 MemoryTotal MemoryAvailable SwapTotal SwapAvailable Eth0InOctets Eth0OutOctets Eth1InOctets Eth1OutOctets
QNS04	CpuUser CpuSystem CpuIdle LoadAverage1 LoadAverage5 LoadAverage15 MemoryTotal MemoryAvailable SwapTotal SwapAvailable Eth0InOctets Eth0OutOctets Eth1InOctets Eth1OutOctets

Component	Information
Portal01	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets
Portal02	CpuUser
	CpuSystem
	CpuIdle
	LoadAverage1
	LoadAverage5
	LoadAverage15
	MemoryTotal
	MemoryAvailable
	SwapTotal
	SwapAvailable
	Eth0InOctets
	Eth0OutOctets
	Eth1InOctets
	Eth1OutOctets

Details of Monitoring and Trending

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)
```

Key Performance Indicators (KPI)

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 KPIs

In this table, the components listed on the left provide the information listed on the right.

Component	Information
LB01/LB02	PCRFProxyExternalCurrentSessions
PortalLB01/PortalLB02	PortalProxyExternalCurrentSessions
PCRFClient01/PCRFClient02	----------
SessionMgr01/SessionMgr02	----------
QNS01 QNS02 QNS03 QNS04	CreateEntryAvgTime CreateEntrySuccess DeleteEntryAvgTime DeleteEntrySuccess GetSessionActionAvgTime GetSessionActionSuccess LockSessionActionAvgTime
	LockSessionActionSuccess PushQuotaAvgTime PushQuotaSuccess QuotaCalculationAvgTime QuotaCalculationSuccess QuotaDepletedRequestAvgTime

Component	Information
	QuotaDepletedRequestSucccess
	RadiusAccountingMessageAvgTime
	RadiusAccountingMessageSuccess
	RetrieveSumAvPairAvgTime
	RetrieveSumAvPairSuccess
	PolicyCount
	QueueSize
	FailedEnqueueCount
	ErrorCount
	SessionCount
Portal01 Portal02	----------

Details of KPIs

The following information is available, and is listed per component. 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 kpi53QNS01CreateEntryAvgTime(2)
| +-- -R-- Integer32 kpi53QNS01CreateEntrySuccess(3)
| +-- -R-- Integer32 kpi53QNS01DeleteEntryAvgTime(4)
| +-- -R-- Integer32 kpi53QNS01DeleteEntrySuccess(5)
| +-- -R-- Integer32 kpi53QNS01GetSessionActionAvgTime(6)
| +-- -R-- Integer32 kpi53QNS01GetSessionActionSuccess(7)
| +-- -R-- Integer32 kpi53QNS01LockSessionActionAvgTime(8)
| +-- -R-- Integer32 kpi53QNS01LockSessionActionSuccess(9)
| +-- -R-- Integer32 kpi53QNS01PushQuotaAvgTime(10)
| +-- -R-- Integer32 kpi53QNS01PushQuotaSuccess(11)
| +-- -R-- Integer32 kpi53QNS01QuotaCalculationAvgTime(12)
| +-- -R-- Integer32 kpi53QNS01QuotaCalculationSuccess(13)
| +-- -R-- Integer32 kpi53QNS01QuotaDepletedRequestAvgTime(14)
| +-- -R-- Integer32 kpi53QNS01QuotaDepletedRequestSucccess(15)
| +-- -R-- Integer32 kpi53QNS01RadiusAccountingMessageAvgTime(16)
| +-- -R-- Integer32 kpi53QNS01RadiusAccountingMessageSuccess(17)
| +-- -R-- Integer32 kpi53QNS01RetrieveSumAvPairAvgTime(18)
| +-- -R-- Integer32 kpi53QNS01RetrieveSumAvPairSuccess(19)
| +-- -R-- Integer32 kpi53QNS01PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS01QueueSize(21)
| +-- -R-- Integer32 kpi53QNS01FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS01ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS01SessionCount(24)
+--broadhopProductsQNSKPI53QNS02(42)
| |
| +-- -R-- Integer32 kpi53QNS02CreateEntryAvgTime(2)
| +-- -R-- Integer32 kpi53QNS02CreateEntrySuccess(3)
```

```
| +-- -R-- Integer32 kpi53QNS02DeleteEntryAvgTime(4)
| +-- -R-- Integer32 kpi53QNS02DeleteEntrySuccess(5)
| +-- -R-- Integer32 kpi53QNS02GetSessionActionAvgTime(6)
| +-- -R-- Integer32 kpi53QNS02GetSessionActionSuccess(7)
| +-- -R-- Integer32 kpi53QNS02LockSessionActionAvgTime(8)
| +-- -R-- Integer32 kpi53QNS02LockSessionActionSuccess(9)
| +-- -R-- Integer32 kpi53QNS02PushQuotaAvgTime(10)
| +-- -R-- Integer32 kpi53QNS02PushQuotaSuccess(11)
| +-- -R-- Integer32 kpi53QNS02QuotaCalculationAvgTime(12)
| +-- -R-- Integer32 kpi53QNS02QuotaCalculationSuccess(13)
| +-- -R-- Integer32 kpi53QNS02QuotaDepletedRequestAvgTime(14)
| +-- -R-- Integer32 kpi53QNS02QuotaDepletedRequestSucccess(15)
| +-- -R-- Integer32 kpi53QNS02RadiusAccountingMessageAvgTime(16)
| +-- -R-- Integer32 kpi53QNS02RadiusAccountingMessageSuccess(17)
| +-- -R-- Integer32 kpi53QNS02RetrieveSumAvPairAvgTime(18)
| +-- -R-- Integer32 kpi53QNS02RetrieveSumAvPairSuccess(19)
| +-- -R-- Integer32 kpi53QNS02PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS02QueueSize(21)
| +-- -R-- Integer32 kpi53QNS02FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS02ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS02SessionCount(24)
+--broadhopProductsQNSKPI53QNS03(43)
| |
| +-- -R-- Integer32 kpi53QNS03CreateEntryAvgTime(2)
| +-- -R-- Integer32 kpi53QNS03CreateEntrySuccess(3)
| +-- -R-- Integer32 kpi53QNS03DeleteEntryAvgTime(4)
| +-- -R-- Integer32 kpi53QNS03DeleteEntrySuccess(5)
| +-- -R-- Integer32 kpi53QNS03GetSessionActionAvgTime(6)
| +-- -R-- Integer32 kpi53QNS03GetSessionActionSuccess(7)
| +-- -R-- Integer32 kpi53QNS03LockSessionActionAvgTime(8)
| +-- -R-- Integer32 kpi53QNS03LockSessionActionSuccess(9)
| +-- -R-- Integer32 kpi53QNS03PushQuotaAvgTime(10)
| +-- -R-- Integer32 kpi53QNS03PushQuotaSuccess(11)
| +-- -R-- Integer32 kpi53QNS03QuotaCalculationAvgTime(12)
| +-- -R-- Integer32 kpi53QNS03QuotaCalculationSuccess(13)
| +-- -R-- Integer32 kpi53QNS03QuotaDepletedRequestAvgTime(14)
| +-- -R-- Integer32 kpi53QNS03QuotaDepletedRequestSucccess(15)
| +-- -R-- Integer32 kpi53QNS03RadiusAccountingMessageAvgTime(16)
```

| +-- -R-- Integer32 kpi53QNS03RadiusAccountingMessageSuccess(17)
| +-- -R-- Integer32 kpi53QNS03RetrieveSumAvPairAvgTime(18)
| +-- -R-- Integer32 kpi53QNS03RetrieveSumAvPairSuccess(19)
| +-- -R-- Integer32 kpi53QNS03PolicyCount(20)
| +-- -R-- Integer32 kpi53QNS03QueueSize(21)
| +-- -R-- Integer32 kpi53QNS03FailedEnqueueCount(22)
| +-- -R-- Integer32 kpi53QNS03ErrorCount(23)
| +-- -R-- Integer32 kpi53QNS03SessionCount(24)
+--broadhopProductsQNSKPI53QNS04(44)
|
+-- -R-- Integer32 kpi53QNS04CreateEntryAvgTime(2)
+-- -R-- Integer32 kpi53QNS04CreateEntrySuccess(3)
+-- -R-- Integer32 kpi53QNS04DeleteEntryAvgTime(4)
+-- -R-- Integer32 kpi53QNS04DeleteEntrySuccess(5)
+-- -R-- Integer32 kpi53QNS04GetSessionActionAvgTime(6)
+-- -R-- Integer32 kpi53QNS04GetSessionActionSuccess(7)
+-- -R-- Integer32 kpi53QNS04LockSessionActionAvgTime(8)
+-- -R-- Integer32 kpi53QNS04LockSessionActionSuccess(9)
+-- -R-- Integer32 kpi53QNS04PushQuotaAvgTime(10)
+-- -R-- Integer32 kpi53QNS04PushQuotaSuccess(11)
+-- -R-- Integer32 kpi53QNS04QuotaCalculationAvgTime(12)
+-- -R-- Integer32 kpi53QNS04QuotaCalculationSuccess(13)
+-- -R-- Integer32 kpi53QNS04QuotaDepletedRequestAvgTime(14)
+-- -R-- Integer32 kpi53QNS04QuotaDepletedRequestSucccess(15)
+-- -R-- Integer32 kpi53QNS04RadiusAccountingMessageAvgTime(16)
+-- -R-- Integer32 kpi53QNS04RadiusAccountingMessageSuccess(17)
+-- -R-- Integer32 kpi53QNS04RetrieveSumAvPairAvgTime(18)
+-- -R-- Integer32 kpi53QNS04RetrieveSumAvPairSuccess(19)
+-- -R-- Integer32 kpi53QNS04PolicyCount(20)
+-- -R-- Integer32 kpi53QNS04QueueSize(21)
+-- -R-- Integer32 kpi53QNS04FailedEnqueueCount(22)
+-- -R-- Integer32 kpi53QNS04ErrorCount(23)
+-- -R-- Integer32 kpi53QNS04SessionCount(24)

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

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
```

If threshold crosses, snmpd throws a trap to LBVIP02 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.

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.

Name	Facility Description	Severity	Operational Action
HighLoad	Operating System	Warning	Not Necessary
	Current system load (average 1 minute) has passed a designated threshold.		
HighLoad	Operating System	Alert	Immediate
	Current system load (average 15 minutes) has passed a designated threshold. Log in and review system status to determine cause. This cause might be due to usage growth, a runaway process, or other causes, but the fact that the load is very high over a long period of time means you should investigate.		
DiskFull	Operating System	Warning	Hours or days
	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.		
DiskFull	Operating System	Alert	Immediate
	Current disk usage has passed a designated threshold. This situation needs immediate access. Should the disk fill up entirely the system may cease to function properly and data corruption may occur.		
LowMemory	Operating System	Warning	Not Necessary
	Current memory usage has passed a designated threshold. This is a warning.		
LowMemory	Operating System	Alert	Immediate
	Current memory usage has passed a designated threshold. This might not signal a persistent issue, but memory is low and the device should be accessed and reviewed for cause.		
LowSwap	Operating System	Warning	Hours or days
	Current swap usage has passed a designated threshold. This is a warning.		
LowSwap	Operating System	Alert	Immediate
	Current swap usage has passed a designated threshold. This might not signal a persistent issue, but memory is low and the device should be accessed and reviewed for cause.		
InterfaceDown	Operating System	Alert	Immediate
	A networking interface on a component has just gone down. This might signal that an administrator is on the system for maintenance, but generally all interfaces are critical to system operation. The devices should be accessed and reviewed for cause.		

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. When trap is generated, it is thrown to LBVIP02 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.

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	Facility	Severity	Operation Action
Memcached InternalError	Application	Error	Minutes to hours
	The memcached application has generated an internal error. System operation continues without interruption of core services, but please investigate to determine the cause and if this is a symptom of a larger issue.		
Memcached ConnectError	Application	Critical	Immediate
	A connection cannot be established to memcached application. System operation continues without interruption of core services, but please investigate to determine the cause and if this is a symptom of a larger issue.		
LicensedSessionCreation	Application	Warning	Days
	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.		

Name	Facility	Severity	Operation Action
LicensedSessionCreation	Operating System	Critical	Immediate
	A predefined threshold of sessions covered by licensing has been passed. The current amount of sessions is very close to the licensed limit. Once passed, the system ceases to operate for new sessions. You must obtain additional licensing.		
InvalidLicense	Application	Emergency	Immediate
	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.		
Policy Configuration	Application	Error	Hours to Days
	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	Critical	Immediate
	The reported session management node is unavailable. This should not immediately affect core services, but may degrade service performance.		
JMSConnection Error	Application	Error	Minutes to hours
	Internal services cannot connect to a required Java JMS queue. Although retry logic and recovery is available, and core system functions should continue, investigate and remediate the root cause soon.		
PoliciesNot Configured	Application	Emergency	Immediate
	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.		
Radius ConnectionError	Application	Critical	Immediate
	A connection cannot be made to the RADIUS server reported. While generally other RADIUS servers are available and the system attempts to continue service, take immediate action to isolate and resolve the issue.		
ApplicationStartError Alert	Application	Alert	Immediate
	The reported application required on the system cannot start. This may not cause interruption of core system functionality, but does require investigation and manual restarting.		

Platform Related Alarms

Platform related alarms are described in the following topics:

- Supported Alarms
- Alarms Generated through Scripts

Supported Alarms

CPS supported alarms are shown in the following list:

Name	Feature	Severity	Message Text	Default Value
Disk Full: This alarm gets generated for following file system: 1. / 2. /var 3. /home 4. /boot 5. /opt	Operating System	Warning	DiskFullAlert	10
	Condition Triggered: 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.			
	Procedure: **Step 1** In /etc/snmp/snmpd.conf, set "disk / 90%". (So when disk remaining is 90% ie. Disk occupied is 10%, alarm is generated.) **Step 2** service snmpd restart **Step 3** tail -f /var/log/snmp/trap **Step 4** trap have messages like :dskErrorMsg.1 = STRING: /: less than 90% free (= 100%)			
Disk Full Clear: This alarm gets generated for following file system: 1. / 2. /var 3. /home 4. /boot 5. /opt	Operating System	Clear	DiskFullClear	-
	Condition Triggered: Current Disk usage has recovered the designated Threshold.			
	Procedure: **Step 1** In /etc/snmp/snmpd.conf, set "disk / X%". (X should just less than actual remaining space. For example, if drive / is 25% full, put 74% as value of X). **Step 2** service snmpd restart **Step 3** Now dump a big file which consumes at least 2-3 % space on drive /. This generates diskful alarm first. **Step 4** Then delete this file. This generates clear alarm. **Step 5** tail -f /var/log/snmp/trap to monitor the alarms generated.			

Name	Feature	Severity	Message Text	Default Value
Load Average of local system: The alarm gets generated for 1 minute 5 minute 15 minute Average	Operating System	Warning (1, 5 minutes) Alert (15 minutes)	HighLoadAlert	4
	Condition Triggered: Current CPU load is more than configured threshold for 1/5/15 minutes.			
	Procedure: **Step 1** In /etc/snmp/snmpd.conf, set "load 1 1 1". (first digit corresponds to average 1 min load. Second digit is for 5 minutes average load. Third is for 15 mins. When it crosses 1 %, alarm is generated.) **Step 2** service snmpd restart **Step 3** tail -f /var/log/snmp/trap **Step 4** trap have message like 1 min Load Average too high (= 1.41)			
Load Average Clear of local system	Operating System	Clear	HighLoadClear	-
	Conditions Triggered: Current CPU load has recovered from more than configured threshold.			
	Procedure: **Step 1** In /etc/snmp/snmpd.conf, set "load 1 1 1". (first digit corresponds to average 1 min load. Second digit is for 5 minutes average load. Third is for 15 mins. When it crosses 1 %, alarm is generated.) **Step 2** service snmpd restart **Step 3** tail -f /var/log/snmp/trap **Step 4** trap have message like 1 min Load Average too high (= 1.41) **Step 5** Now wait till load comes below 1 % OR write a script which runs in infinite loop and create some load on CPU. Killing this script reduces the load and generates clear trap.			

Name	Feature	Severity	Message Text	Default Value
Low Swap memory alarm	Operating System	Warning	LowSwapAlert	102400
	Conditions Triggered: Current swap usage has passed a designated threshold.			
	Procedure: **Step 1** To find the Swap allocated, used and remaining, use command cat /proc/meminfo. **Step 2** In /etc/snmp/snmpd.conf, set "swap X". (So when swap remaining is X kB, alarm is generated. X should be slightly greater than actual remaining swap. For example, if swap remaining is 1000,000 kB, X should be 1000,030 KB) **Step 3** service snmpd restart **Step 4** tail -f /var/log/snmp/trap **Step 5** trap have message mentioning swap.			
Low Swap memory clear	Operating System	Clear	LowSwapClear	-
	Conditions Triggered: Current swap usage has recovered a designated threshold.			
	Procedure: **Step 1** To find the Swap allocated, used and remaining, use command cat /proc/meminfo. **Step 2** In /etc/snmp/snmpd.conf, set "swap X". (So when swap remaining is X kB, alarm is generated. X should be slightly greater than actual remaining swap. For example, if swap remaining is 1000,000 kB, X should be 1000,030 KB) **Step 3** service snmpd restart **Step 4** tail -f /var/log/snmp/trap **Step 5** trap have message mentioning swap.			
Interface Down Alarm: This alarm gets generated for all physical interface attached to the system.	Operating System	Warning	<Interface Name> is Down	-
	Conditions Triggered: Not able to connect or ping to the interface **Command:** Ifconfig eth1 down			

Name	Feature	Severity	Message Text	Default Value
Interface Up Alarm: This alarm gets generated for all physical interface attached to the system.	Operating System	Clear	<Interface Name> is Up	-
	Conditions Triggered: Able to ping or connect to interface **Command:** Ifconfig eth1 up			

Alarms Generated through Scripts

Alarms generated through scripts are shown in the following list:

- Every 1 minute

Error Code	Feature	Severity	Message Text
7001	Core	Critical	HA Failover
	Condition Triggered: Primary member of replica is down and taken over by other primary member of same replica set.		
	Steps to reproduce: Shutdown primary member VM like sessionMgr01 and SessionMgr02.		
7002	Core	Critical	Geo Failover
	Condition Triggered: Primary Member of Replica set is down and taken over by other member of secondary member of same replica set.		
	Steps to reproduce: Shutdown sessionMgr VMs on primary site.		
7003	Core	Critical	All members of replicas are down
	Conditions Triggered: Not able to connect all members of replica set.		
	Steps to reproduce: **Step 1** Get all members of replica set from /etc/broadhop/mongoconfig.cfg. **Step 2** Go to each sessionMgr from there and cd /etc/init.d **Step 3** service sessionmgr-27717 stop		
7004	Core	Critical	No Primary Member Found
	Conditions Triggered: In replica set if there is no primary member configured.		
	Steps to reproduce: **Step 1** Shutdown all sessionMgr VMs mentioned in replica set. **Step 2** Members of replica set can be found by following command: [root@pcrfclient01 ~]# cat /etc/broadhop/mongoConfig.cfg		

- Every 5 minutes

Error Code	Feature	Severity	Message Text
7005	Core	Critical	Secondary DB Member Down
	Condition Triggered: In Replica Set if Secondary member not able to connect.		
	Steps to reproduce: Get secondary members in /etc/broadhop/mongoConfig.cfg on pcrfclient01. Make these VMs down one by one.		
7006	Core	Critical	Arbiter Down
	Condition Triggered: In replica set Not able to connect to configured arbiter.		
	Steps to reproduce: Get arbiter list as follows: [root@pcrfclient01 ~]# cat /etc/broadhop/mongoConfig.cfg | grep -i arbit Shutdown VMs corresponding to it.		
7007	Core	Critical	Config Server is Down
	Conditions Triggered: In replica set not able to connect to Configured Config Server.		
	Steps to reproduce: Get config servers by following command: [root@pcrfclient01 ~]# cat /etc/broadhop/mongoConfig.cfg | grep -i config Shutdown these VMs one after another.		
8001	Core	Critical	VM Down
	Conditions Triggered: If not able to ping to VM (This alarms gets generated for all VMs configured inside /etc/hosts of lb)		
	Steps to reproduce: Note all VMs in /etc/hots. Shutdown them one after another.		
9001	Core	Critical	Virtual Interface Down
	Conditions Triggered: Not able to ping the virtual Interface. This alarm gets generated for lbvip01, lbvip02.		
	Steps to reproduce: Command : Ifconfig eth1:0 down		

Diagnostic Nodes, Messages and Errors

Diagnostic nodes, messages and errors are shown in the following list:

Error Code	Feature	Severity	Message Text	Conditions Triggered	Class
1	Core	Normal	QNS Server is Alive.	Always when CPS server is running.	DiagnosticNode
2	Core	Normal	Memcached server is operational.	Generated if the system is able to connect to memcached server and write to it.	DiagnosticNode

Error Code	Feature	Severity	Message Text	Conditions Triggered	Class
2	Core	Error	Memcached server is in error	Generated if the system is unable to connect to memcached server or unable to write to it.	DiagnosticNode
2	Core	Error	Memcached server is in error: %s	Generated if attempting to connect to or write to the memcached server causes an exception. %s is the exception that occurred.	DiagnosticNode
3	Core	Error	Feature %s is unable to start. Error %s	Generated if an installed feature cannot start.	DiagnosticNode
4	Core	Normal	Session creation allowed. %s % used.	Generated if session creation is under 90% of license limit.	DiagnosticNode
4	Core	Warning	Session creation allowed. %s % used.	Generated if session creation is allowed and session count > 90% of allowed license limit.	DiagnosticNode
4	Core	Error	Session creation is not allowed.	Generated if license is not installed or session count has reached allowed license level.	DiagnosticNode
5	Core	Normal	Valid license found: %s	License is valid.	DiagnosticNode
5	Core	Error	Invalid license found: %s	License is not valid.	DiagnosticNode
6	Core	Normal	Last policy configuration was successful	Policy Engine was successfully initialized and policies were configured.	PolicyConfiguration
6	Core	Error	Last policy configuration failed with the following message: %s	Policy Engine is not initialized or an exception has occurred impeding policy configuration.	PolicyConfiguration
7	Core	Normal	%s session management node is enabled	Session management node is enabled. %s is the IP and Port of the node.	ShardRunnerSupport
7	Core	Error	%s session management node is not available	Session management node is not available. %s is the IP and Port of the node.	ShardRunnerSupport
8	Core	Normal	Current node is expiration manager.	When current node is responsible for expiring sessions.	ExpirationManager

Error Code	Feature	Severity	Message Text	Conditions Triggered	Class
9	Core	Normal	JMS Operating Normally	When the JMS has not started, or the Connection was initially successful.	Jms
9	Core	Error	Unable to connect to JMS Server	When the JMS attempted an initial connection and it has not yet succeeded.	Jms
1000	Core	Normal	Policies successfully configured	Policies successfully configured on init (NOT a diagnostic node).	PolicyConfiguration
1000	Core	Error	Policies not configured	Policies not configured on init (NOT a diagnostic node).	PolicyConfiguration
10000	Radius	Normal	Successfully pinged %s in %n ms	Radius server successfully pinged.	RadiusDiagnostics
10000	Radius	Error	Failed ping on %s message: %s1	Radius server ping failed.	RadiusDiagnostics
11000	Arbor	Normal	Connection to SM is available	Arbor device is available.	SMInvoker
11000	Arbor	Warning	Connection to SM - running in simulation mode	Code is configured to skip calls to Arbor device.	SMInvoker
11000	Arbo	Error	No connection to SM is available.	Arbor device is not available.	SMInvoker
12000	Sum	Error	Sum connection is unavailable: %s	No external API manager is available.	SumApiManager
12000	Sum	Warning	Connection is available. 5 retrievals in %n ms.	Response time is greater than 100 ms.	SumApiManager
12000	Sum	Normal	Connection is available. 5 retrievals in %n ms.	Response time is less than 100 ms.	SumApiManager
13000	Diameter	Normal	Stack %s running.	Diameter stack is running.	DiameterStackManager
14000	Reporting	Normal	CSV replication is running for %s	The virtual replication server is running for the given server name.	CsvReplicationRunner
14001	Reporting	Normal	CSV replication is not running for %s	The virtual replication server is not running for the given server name.	CsvReplicationRunner

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 two pieces:

- Configuration for SNMP gets and walks
- Configuration for Notifications (traps)
- 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 and the patch 10042012a_snmp_framework_patch has been applied.

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, restart the snmptrapd service to enable changes.

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.

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.