JUNIPER NETWORKS

Junos® OS

Network Management Administration Guide for Routing Devices

Release
13.2

Published: 2013-07-19

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Junos® OS Network Management Administration Guide for Routing Devices
13.2

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at http://www.juniper.net/support/eula.html. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

List of Figures

List of Tables

About the Documentation

Documentation and Release Notes

To obtain the most current version of all Juniper Networks® technical documentation, see the product documentation page on the Juniper Networks website at http://www.juniper.net/techpubs/.

If the information in the latest release notes differs from the information in the documentation, follow the product Release Notes.

Juniper Networks Books publishes books by Juniper Networks engineers and subject matter experts. These books go beyond the technical documentation to explore the nuances of network architecture, deployment, and administration. The current list can be viewed at http://www.juniper.net/books.

Supported Platforms

For the features described in this document, the following platforms are supported:

- ACX Series
- M Series
- MX Series
- T Series
- PTX Series

Using the Examples in This Manual

If you want to use the examples in this manual, you can use the **load merge** or the **load merge relative** command. These commands cause the software to merge the incoming

configuration into the current candidate configuration. The example does not become active until you commit the candidate configuration.

If the example configuration contains the top level of the hierarchy (or multiple hierarchies), the example is a *full example*. In this case, use the **load merge** command.

If the example configuration does not start at the top level of the hierarchy, the example is a *snippet*. In this case, use the **load merge relative** command. These procedures are described in the following sections.

Merging a Full Example

To merge a full example, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration example into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following configuration to a file and name the file **ex-script.conf**. Copy the **ex-script.conf** file to the **/var/tmp** directory on your routing platform.

   ```
   system {
     scripts {
       commit {
         file ex-script.xsl;
       }
     }
   }
   interfaces {
     fxp0 {
       disable;
       unit 0 {
         family inet {
           address 10.0.0.1/24;
         }
       }
     }
   }
   ```

2. Merge the contents of the file into your routing platform configuration by issuing the **load merge** configuration mode command:

   ```
   [edit]
   user@host# load merge /var/tmp/ex-script.conf
   load complete
   ```

Merging a Snippet

To merge a snippet, follow these steps:

1. From the HTML or PDF version of the manual, copy a configuration snippet into a text file, save the file with a name, and copy the file to a directory on your routing platform.

   For example, copy the following snippet to a file and name the file **ex-script-snippet.conf**. Copy the **ex-script-snippet.conf** file to the **/var/tmp** directory on your routing platform.

```
commit {
  file ex-script-snippet.xsl; }
```

2. Move to the hierarchy level that is relevant for this snippet by issuing the following configuration mode command:

```
[edit]
user@host# edit system scripts
[edit system scripts]
```

3. Merge the contents of the file into your routing platform configuration by issuing the **load merge relative** configuration mode command:

```
[edit system scripts]
user@host# load merge relative /var/tmp/ex-script-snippet.conf
load complete
```

For more information about the **load** command, see the *CLI User Guide*.

Documentation Conventions

Table 1 on page xix defines notice icons used in this guide.

Table 1: Notice Icons

Icon	Meaning	Description
	Informational note	Indicates important features or instructions.
	Caution	Indicates a situation that might result in loss of data or hardware damage.
	Warning	Alerts you to the risk of personal injury or death.
	Laser warning	Alerts you to the risk of personal injury from a laser.

Table 2 on page xix defines the text and syntax conventions used in this guide.

Table 2: Text and Syntax Conventions

Convention	Description	Examples
Bold text like this	Represents text that you type.	To enter configuration mode, type the**configure** command: user@host> **configure**
`Fixed-width text like this`	Represents output that appears on the terminal screen.	`user@host>` **show chassis alarms** `No alarms currently active`

Table 2: Text and Syntax Conventions *(continued)*

Convention	Description	Examples
Italic text like this	• Introduces or emphasizes important new terms. • Identifies book names. • Identifies RFC and Internet draft titles.	• A policy *term* is a named structure that defines match conditions and actions. • *Junos OS System Basics Configuration Guide* • RFC 1997, *BGP Communities Attribute*
Italic text like this	Represents variables (options for which you substitute a value) in commands or configuration statements.	Configure the machine's domain name: [edit] root@# **set system domain-name** *domain-name*
Text like this	Represents names of configuration statements, commands, files, and directories; configuration hierarchy levels; or labels on routing platform components.	• To configure a stub area, include the **stub** statement at the **[edit protocols ospf area area-id]** hierarchy level. • The console port is labeled **CONSOLE**.
< > (angle brackets)	Enclose optional keywords or variables.	**stub <default-metric** ***metric*****>;**
| (pipe symbol)	Indicates a choice between the mutually exclusive keywords or variables on either side of the symbol. The set of choices is often enclosed in parentheses for clarity.	**broadcast | multicast** **(*string1* | *string2* | *string3*)**
# (pound sign)	Indicates a comment specified on the same line as the configuration statement to which it applies.	**rsvp { # Required for dynamic MPLS only**
[ ] (square brackets)	Enclose a variable for which you can substitute one or more values.	**community name members [** ***community-ids*** **]**

Indention and braces ({ })	Identify a level in the configuration hierarchy.	[edit]
		routing-options {
		  static {
		  route default {
		  nexthop *address*;
		  retain;
		  }
		  }
		}

; (semicolon)	Identifies a leaf statement at a configuration hierarchy level.	
GUI Conventions		
Bold text like this	Represents graphical user interface (GUI) items you click or select.	• In the Logical Interfaces box, select **All Interfaces**. • To cancel the configuration, click **Cancel**.
> (bold right angle bracket)	Separates levels in a hierarchy of menu selections.	In the configuration editor hierarchy, select **Protocols>Ospf**.

Documentation Feedback

We encourage you to provide feedback, comments, and suggestions so that we can improve the documentation. You can send your comments to techpubs-comments@juniper.net, or fill out the documentation feedback form at https://www.juniper.net/cgi-bin/docbugreport/ . If you are using e-mail, be sure to include the following information with your comments:

- Document or topic name
- URL or page number
- Software release version (if applicable)

Requesting Technical Support

Technical product support is available through the Juniper Networks Technical Assistance Center (JTAC). If you are a customer with an active J-Care or JNASC support contract, or are covered under warranty, and need post-sales technical support, you can access our tools and resources online or open a case with JTAC.

- JTAC policies—For a complete understanding of our JTAC procedures and policies, review the *JTAC User Guide* located at http://www.juniper.net/us/en/local/pdf/resource-guides/7100059-en.pdf.
- Product warranties—For product warranty information, visit http://www.juniper.net/support/warranty/.
- JTAC hours of operation—The JTAC centers have resources available 24 hours a day, 7 days a week, 365 days a year.

Self-Help Online Tools and Resources

For quick and easy problem resolution, Juniper Networks has designed an online self-service portal called the Customer Support Center (CSC) that provides you with the following features:

- Find CSC offerings: http://www.juniper.net/customers/support/
- Search for known bugs: http://www2.juniper.net/kb/
- Find product documentation: http://www.juniper.net/techpubs/
- Find solutions and answer questions using our Knowledge Base: http://kb.juniper.net/
- Download the latest versions of software and review release notes: http://www.juniper.net/customers/csc/software/
- Search technical bulletins for relevant hardware and software notifications: https://www.juniper.net/alerts/

- Join and participate in the Juniper Networks Community Forum: http://www.juniper.net/company/communities/
- Open a case online in the CSC Case Management tool: http://www.juniper.net/cm/

To verify service entitlement by product serial number, use our Serial Number Entitlement (SNE) Tool: https://tools.juniper.net/SerialNumberEntitlementSearch/

Opening a Case with JTAC

You can open a case with JTAC on the Web or by telephone.

- Use the Case Management tool in the CSC at http://www.juniper.net/cm/.
- Call 1-888-314-JTAC (1-888-314-5822 toll-free in the USA, Canada, and Mexico).

For international or direct-dial options in countries without toll-free numbers, see http://www.juniper.net/support/requesting-support.html.

PART 1

Overview

CHAPTER 1

Network Management Technologies and Features

Understanding Device Management Functions in Junos OS

After you have installed a device into your network, you need to manage the device within your network. Device management can be divided into five tasks:

- Fault management—Monitor the device; detect and fix faults.
- Configuration management—Configure device attributes.
- Accounting management—Collect statistics for accounting purposes.
- Performance management—Monitor and adjust device performance.
- Security management—Control device access and authenticate users.

The Junos® operating system (Junos OS) network management features work in conjunction with an operations support system (OSS) to manage the devices within the network. Junos OS can assist you in performing these management tasks, as described in Table 3 on page 4.

Table 3: Device Management Features in Junos OS

Task	Junos OS Feature
Fault management	Monitor and see faults using: • Operational mode commands—For more information about operational mode commands, see the *Junos OS Operational Mode Commands*, *Junos OS Operational Mode Commands*, and *Junos OS Operational Mode Commands*. • SNMP MIBs—For more information about SNMP MIBs supported by Junos OS, see ""Standard SNMP MIBs Supported by Junos OS" on page 19" and ""Juniper Networks Enterprise-Specific MIBs" on page 35" in the *SNMP MIBs and Traps Reference*. • Standard SNMP traps—For more information about standard SNMP traps, see the ""Standard SNMP Traps Supported on Devices Running Junos OS" on page 63" in the *SNMP MIBs and Traps Reference*. • Enterprise-specific SNMP traps—For more information about enterprise-specific traps, see ""Juniper Networks Enterprise-Specific SNMP Traps" on page 63" in the *SNMP MIBs and Traps Reference*. • System log messages—For more information about how to configure system log messages, see the *Junos OS Administration Library for Routing Devices*. For more information about how to view system log messages, see the *Junos OS System Log Messages Reference*.
Configuration management	• Configure router attributes using the command-line interface (CLI), the Junos XML management protocol, and the NETCONF XML management protocol. For more information about configuring the router using the CLI, see the *Junos OS Administration Library for Routing Devices*. For more information about configuring the router using the APIs, see the *Junos XML Management Protocol Guide* and *NETCONF XML Management Protocol Guide*. • Configuration Management MIB—For more information about the Configuration Management MIB, see the "*Configuration Management MIB*" in the *SNMP MIBs and Traps Reference*.

Table 3: Device Management Features in Junos OS *(continued)*

Task	Junos OS Feature
Accounting management	Perform the following accounting-related tasks: • Collect statistics for interfaces, firewall filters, destination classes, source classes, and the Routing Engine. For more information about collecting statistics, see "Accounting Options Configuration" on page 168. • Use interface-specific traffic statistics and other counters, available in the Standard Interfaces MIB, Juniper Networks enterprise-specific extensions to the Interfaces MIB, and media-specific MIBs, such as the enterprise-specific ATM MIB. • Use per-ATM virtual circuit (VC) counters, available in the enterprise-specific ATM MIB. For more information about the ATM MIB, see the *SNMP MIBs and Traps Reference*. • Group source and destination prefixes into source classes and destination classes and count packets for those classes. Collect destination class and source class usage statistics. For more information about classes, see "*Destination Class Usage MIB*" and "*Source Class Usage MIB*" in the *SNMP MIBs and Traps Reference*, "Configuring Class Usage Profiles" on page 185, the *Junos OS Network Interfaces Library for Routing Devices*, and the *Routing Policy Feature Guide for Routing Devices*. • Count packets as part of a firewall filter. For more information about firewall filter policies, see ""Juniper Networks Enterprise-Specific MIBs" on page 35" in the *SNMP MIBs and Traps Reference* and the *Routing Policy Feature Guide for Routing Devices*. • Sample traffic, collect the samples, and send the collection to a host running the CAIDA cflowd utility. For more information about CAIDA and cflowd, see the *Routing Policy Feature Guide for Routing Devices*.
Performance management	Monitor performance in the following ways: • Use operational mode commands. For more information about monitoring performance using operational mode commands, see the *Junos OS Operational Mode Commands*. • Use firewall filter. For more information about performance monitoring using firewall filters, see the *Routing Policy Feature Guide for Routing Devices*. • Sample traffic, collect the samples, and send the samples to a host running the CAIDA cflowd utility. For more information about CAIDA and cflowd, see the *Routing Policy Feature Guide for Routing Devices*. • Use the enterprise-specific Class-of-Service MIB. For more information about this MIB, see the "*Class-of-Service MIB*" in the *SNMP MIBs and Traps Reference*.
Security management	Assure security in your network in the following ways: • Control access to the router and authenticate users. For more information about access control and user authentication, see the *Junos OS Administration Library for Routing Devices*. • Control access to the router using SNMPv3 and SNMP over IPv6. For more information, see "Configuring the Local Engine ID" on page 111 and "Tracing SNMP Activity on a Device Running Junos OS" on page 299.

Related Documentation

- Understanding the Integrated Local Management Interface on page 6
- Understanding the SNMP Implementation in Junos OS on page 6
- Understanding Measurement Points, Key Performance Indicators, and Baseline Values on page 78
- Accounting Options Overview on page 81

Understanding the Integrated Local Management Interface

The Integrated Local Management Interface (ILMI) provides a mechanism for Asynchronous Transfer Mode (ATM)-attached devices, such as hosts, routers, and ATM switches, to transfer management information. ILMI provides bidirectional exchange of management information between two ATM interfaces across a physical connection. ILMI information is exchanged over a direct encapsulation of SNMP version 1 (RFC 1157, *A Simple Network Management Protocol*) over ATM Adaptation Layer 5 (AAL5) using a virtual path identifier/virtual channel identifier (VPI/VCI) value (VPI=0, VCI=16).

Junos OS supports only two ILMI MIB variables: **atmfMYIPNmAddress** and **atmfPortMyIfname**. For ATM1 and ATM2 intelligent queuing (IQ) interfaces, you can configure ILMI to communicate directly with an attached ATM switch to enable querying of the switch's IP address and port number.

For more information about configuring ILMI, see the *Junos OS Network Interfaces Library for Routing Devices*. For information about displaying ILMI statistics, see the *Junos OS Operational Mode Commands*. For more information about the ILMI MIB, see the ATM Forum at http://www.atmforum.com/.

Related Documentation

- Understanding Device Management Functions in Junos OS on page 3

Understanding the SNMP Implementation in Junos OS

SNMP enables the monitoring of network devices from a central location. This topic provides an overview of SNMP and describes how SNMP is implemented in Junos® operating system (Junos OS).

This topic includes the following sections:

- SNMP Architecture on page 6
- Junos OS SNMP Agent Features on page 9

SNMP Architecture

The SNMP agent exchanges network management information with SNMP manager software running on a network management system (NMS), or host. The agent responds to requests for information and actions from the manager. The agent also controls access to the agent's MIB, the collection of objects that can be viewed or changed by the SNMP manager.

The SNMP manager collects information about network connectivity, activity, and events by polling managed devices.

Communication between the agent and the manager occurs in one of the following forms:

- **Get**, **GetBulk**, and **GetNext** requests—The manager requests information from the agent; the agent returns the information in a **Get** response message.
- **Set** requests—The manager changes the value of a MIB object controlled by the agent; the agent indicates status in a **Set** response message.
- **Traps** notification—The agent sends traps to notify the manager of significant events that occur on the network device.

This topic contains the following sections:

SNMP MIBs

A MIB is a hierarchy of information used to define managed objects in a network device. The MIB structure is based on a tree structure, which defines a grouping of objects into related sets. Each object in the MIB is associated with an object identifier (OID), which names the object. The "leaf" in the tree structure is the actual managed object instance, which represents a resource, event, or activity that occurs in your network device.

MIBs are either standard or enterprise-specific. Standard MIBs are created by the Internet Engineering Task Force (IETF) and documented in various RFCs. Depending on the vendor, many standard MIBs are delivered with the NMS software. You can also download the standard MIBs from the IETF website, www.ietf.org, and compile them into your NMS, if necessary.

For a list of standard supported MIBs, see "Standard SNMP MIBs Supported by Junos OS" on page 19 in the *SNMP MIBs and Traps Reference* document.

Enterprise-specific MIBs are developed and supported by a specific equipment manufacturer. If your network contains devices that have enterprise-specific MIBs, you must obtain them from the manufacturer and compile them into your network management software.

For a list of Juniper Networks enterprise-specific supported MIBs, see "Juniper Networks Enterprise-Specific MIBs" on page 35 in the *SNMP MIBs and Traps Reference* document.

SNMP Traps and Informs

Routers can send notifications to SNMP managers when significant events occur on a network device, most often errors or failures. SNMP notifications can be sent as traps or inform requests. SNMP traps are unconfirmed notifications. SNMP informs are confirmed notifications.

SNMP traps are defined in either standard or enterprise-specific MIBs. Standard traps are created by the IETF and documented in various RFCs. The standard traps are compiled into the network management software. You can also download the standard traps from the IETF website, www.ietf.org .

For more information about standard traps supported by Junos OS, see "Standard SNMP Traps Supported on Devices Running Junos OS" on page 63 in the *SNMP MIBs and Traps Reference* document.

Enterprise-specific traps are developed and supported by a specific equipment manufacturer. If your network contains devices that have enterprise-specific traps, you must obtain them from the manufacturer and compile them into your network management software.

For more information about enterprise-specific traps supported by Junos OS, see "Juniper Networks Enterprise-Specific SNMP Traps" on page 63 in the *SNMP MIBs and Traps Reference* document. For information about system logging severity levels for SNMP traps, see "System Logging Severity Levels for SNMP Traps" on page 9.

With traps, the receiver does not send any acknowledgment when it receives a trap, and the sender cannot determine if the trap was received. To increase reliability, SNMP informs are supported in SNMPv3. An SNMP manager that receives an inform acknowledges the message with a response. For information about SNMP informs, see "Configuring SNMP Informs" on page 139.

SNMP Trap Queuing

Junos OS supports trap queuing to ensure that traps are not lost because of temporary unavailability of routes. Two types of queues, destination queues and a throttle queue, are formed to ensure delivery of traps and to control the trap traffic.

Junos OS forms a destination queue when a trap to a particular destination is returned because the host is not reachable, and adds the subsequent traps to the same destination to the queue. Junos OS checks for availability of routes every 30 seconds and sends the traps from the destination queue in a round-robin fashion. If the trap delivery fails, the trap is added back to the queue, and the delivery attempt counter and the next delivery attempt timer for the queue are reset. Subsequent attempts occur at progressive intervals of 1 minute, 2 minutes, 4 minutes, and 8 minutes. The maximum delay between the attempts is 8 minutes, and the maximum number of attempts is 10. After 10 unsuccessful attempts, the destination queue and all the traps in the queue are deleted.

Junos OS also has a throttle mechanism to control the number of traps (**throttle threshold**; default value of 500 traps) sent during a particular time period (**throttle interval**; default of 5 seconds) and to ensure consistency in trap traffic, especially when a large number of traps are generated because of interface status changes. The throttle interval period begins when the first trap arrives at the throttle. All traps within the trap threshold are processed, and the traps beyond the threshold limit are queued. The maximum size of trap queues (that is, the throttle queue and the destination queue combined) is 40,000 traps. However, on EX Series switches, the maximum size of the trap queue is 1000 traps. The maximum size of any one queue is 20,000 traps for devices other than EX Series switches. On EX Series switches, the maximum size of one queue is 500 traps. If a trap

is sent from a destination queue when the throttle queue has exceeded the maximum size, the trap is added back to the top of the destination queue, and all subsequent attempts from the destination queue are stopped for a 30-second period, after which the destination queue restarts sending the traps.

NOTE: Users cannot configure Junos OS for trap queuing. Users cannot view any information about trap queues except what is available in the syslog.

System Logging Severity Levels for SNMP Traps

For some traps, when a trap condition occurs, regardless of whether the SNMP agent sends a trap to an NMS, the trap is logged if the system logging is configured to log an event with that system logging severity level. For more information about system logging severity levels, see the *Junos OS Administration Library for Routing Devices* document.

For more information about system logging severity levels for standard traps, see *Standard SNMP Version 1 Traps* and *Standard SNMP Version 2 Traps* in the *SNMP MIBs and Traps Reference* document. For more information about system logging severity levels for enterprise-specific traps, see *Juniper Networks Enterprise-Specific SNMP Version 1 Traps* and *Juniper Networks Enterprise-Specific SNMP Version 2 Traps* in the *SNMP MIBs and Traps Reference* document.

Junos OS SNMP Agent Features

The Junos OS SNMP agent software consists of an SNMP master agent that delegates all SNMP requests to subagents. Each subagent is responsible for the support of a specific set of MIBs.

Junos OS supports the following versions of SNMP:

- SNMPv1—The initial implementation of SNMP that defines the architecture and framework for SNMP.
- SNMPv2c—The revised protocol, with improvements to performance and manager-to-manager communications. Specifically, SNMPv2c implements community strings, which act as passwords when determining who, what, and how the SNMP clients can access the data in the SNMP agent. The community string is contained in SNMP **Get**, **GetBulk**, **GetNext**, and **Set** requests. The agent may require a different community string for **Get**, **GetBulk**, and **GetNext** requests (**read-only** access) than it does for **Set** requests (**read-write** access).
- SNMPv3—The most up-to-date protocol focuses on security. SNMPv3 defines a security model, user-based security model (USM), and a view-based access control model (VACM). SNMPv3 USM provides data integrity, data origin authentication, message replay protection, and protection against disclosure of the message payload. SNMPv3 VACM provides access control to determine whether a specific type of access (read or write) to the management information is allowed.

In addition, the Junos OS SNMP agent software accepts IPv4 and IPv6 addresses for transport over IPv4 and IPv6. For IPv6, Junos OS supports the following features:

- SNMP data over IPv6 networks
- IPv6-specific MIB data
- SNMP agents for IPv6

Related Documentation

- SNMPv3 Overview on page 10
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

SNMPv3 Overview

In contrast to SNMP version 1 (SNMPv1) and SNMP version 2 (SNMPv2), SNMP version 3 (SNMPv3) supports authentication and encryption. SNMPv3 uses the user-based security model (USM) for message security and the view-based access control model (VACM) for access control. USM specifies authentication and encryption. VACM specifies access-control rules.

USM uses the concept of a user for which security parameters (levels of security, authentication, privacy protocols, and keys) are configured for both the agent and the manager. Messages sent using USM are better protected than messages sent with community strings, where passwords are sent in the clear. With USM, messages exchanged between the manager and the agent can have data integrity checking and data origin authentication. USM protects against message delays and message replays by using time indicators and request IDs. Encryption is also available.

To complement the USM, SNMPv3 uses the VACM, a highly granular access-control model for SNMPv3 applications. Based on the concept of applying security policies to the name of the groups querying the agent, the agent decides whether the group is allowed to view or change specific MIB objects. VACM defines collections of data (called views), groups of data users, and access statements that define which views a particular group of users can use for reading, writing, or receiving traps.

Trap entries in SNMPv3 are created by configuring the notify, notify filter, target address, and target parameters. The **notify** statement specifies the type of notification (trap) and contains a single tag. The tag defines a set of target addresses to receive a trap. The notify filter defines access to a collection of trap object identifiers (OIDs). The target address defines a management application's address and other attributes to be used in sending notifications. Target parameters define the message processing and security parameters to be used in sending notifications to a particular management target.

To configure SNMPv3, perform the following tasks:

- Creating SNMPv3 Users on page 118
- Configuring MIB Views on page 109
- Defining Access Privileges for an SNMP Group on page 122

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring SNMP Informs on page 139

Related Documentation

- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

CHAPTER 2

SNMP Remote Operations and Support for Routing Instances

SNMP Remote Operations Overview

A SNMP remote operation is any process on the router that can be controlled remotely using SNMP. Junos OS currently provides support for two SNMP remote operations: the Ping MIB and Traceroute MIB, defined in RFC 2925. Using these MIBs, an SNMP client in the network management system (NMS) can:

- Start a series of operations on a router
- Receive notification when the operations are complete
- Gather the results of each operation

Junos OS also provides extended functionality to these MIBs in the Juniper Networks enterprise-specific extensions **jnxPingMIB** and **jnxTraceRouteMIB**. For more information about **jnxPingMIB** and **jnxTraceRouteMIB**, see the *PING MIB* and *Traceroute MIB* topics in the *SNMP MIBs and Traps Reference*.

This topic covers the following sections:

SNMP Remote Operation Requirements

To use SNMP remote operations, you should be experienced with SNMP conventions. You must also configure Junos OS to allow the use of the remote operation MIBs.

Setting SNMP Views

All remote operation MIBs supported by Junos OS require that the SNMP clients have read-write privileges. The default SNMP configuration of Junos OS does not provide clients with a community string with such privileges.

To set read-write privileges for an SNMP community string, include the following statements at the **[edit snmp]** hierarchy level:

```
[edit snmp]
community community-name {
    authorization authorization;
    view view-name;
}
view view-name {
    oid object-identifier (include | exclude);
}
```

Example: Setting SNMP Views

To create a community named **remote-community** that grants SNMP clients read-write access to the Ping MIB, **jnxPing** MIB, Traceroute MIB, and **jnxTraceRoute** MIB, include the following statements at the **[edit snmp]** hierarchy level:

```
snmp {
    view remote-view {
        oid 1.3.6.1.2.1.80 include; # pingMIB
        oid 1.3.6.1.4.1.2636.3.7 include; # jnxPingMIB
        oid 1.3.6.1.2.1.81 include; # traceRouteMIB
        oid 1.3.6.1.4.1.2636.3.8 include; # jnxTraceRouteMIB
    }
    community remote-community {
        view remote-view;
        authorization read-write;
    }
}
```

For more information about the **community** statement, see "Configuring the SNMP Community String" on page 95 and **community (SNMP)**.

For more information about the **view** statement, see "Configuring MIB Views" on page 109, **view (Associating a MIB View with a Community)**, and **view (Configuring a MIB View)**.

Setting Trap Notification for Remote Operations

In addition to configuring the remote operations MIB for trap notification, you must also configure Junos OS. You must specify a target host for remote operations traps.

To configure trap notification for SNMP remote operations, include the **categories** and **targets** statements at the **[edit snmp trap-group *group-name*]** hierarchy level:

```
[edit snmp trap-group group-name]
    categories {
        category;
    }
    targets {
```

```
        address;
    }
}
```

Example: Setting Trap Notification for Remote Operations

Specify **172.17.12.213** as a target host for all remote operation traps:

```
snmp {
    trap-group remote-traps {
        categories remote-operations;
        targets {
            172.17.12.213;
        }
    }
}
```

For more information about trap groups, see "Configuring SNMP Trap Groups" on page 104.

Using Variable-Length String Indexes

All tabular objects in the remote operations MIBs supported by Junos OS are indexed by two variables of type **SnmpAdminString**. For more information about **SnmpAdminString**, see RFC 2571.

Junos OS does not handle **SnmpAdminString** any differently from the octet string variable type. However, the indexes are defined as variable length. When a variable length string is used as an index, the length of the string must be included as part of the object identifier (OID).

Example: Set Variable-Length String Indexes

To reference the **pingCtlTargetAddress** variable of a row in **pingCtlTable** where **pingCtlOwnerIndex** is **bob** and **pingCtlTestName** is **test**, use the following object identifier (OID):

```
pingMIB.pingObjects.pingCtlTable.pingCtlEntry.pingCtlTargetAddress."bob"."test"
1.3.6.1.2.1.80.1.2.1.4.3.98.111.98.4.116.101.115.116
```

For more information about the definition of the Ping MIB, see RFC 2925.

Enabling Logging

The SNMP error code returned in response to SNMP requests can only provide a generic description of the problem. The error descriptions logged by the remote operations process can often provide more detailed information about the problem and help you to solve the problem faster. This logging is not enabled by default. To enable logging, include the **flag general** statement at the **[edit snmp traceoptions]** hierarchy level:

```
[edit]
snmp {
    traceoptions {
        flag general;
    }
}
```

For more information about traceoptions, see "Tracing SNMP Activity on a Device Running Junos OS" on page 299.

If the remote operations process receives an SNMP request that it cannot accommodate, the error is logged in the **/var/log/rmopd** file. To monitor this log file, issue the **monitor start rmopd** command in operational mode of the command-line interface (CLI).

Related Documentation

- Using the Ping MIB for Remote Monitoring Devices Running Junos OS on page 77
- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77

Understanding SNMP Support for Routing Instances

Junos OS enables SNMP managers for all routing instances to request and manage SNMP data related to the corresponding routing instances and logical system networks.

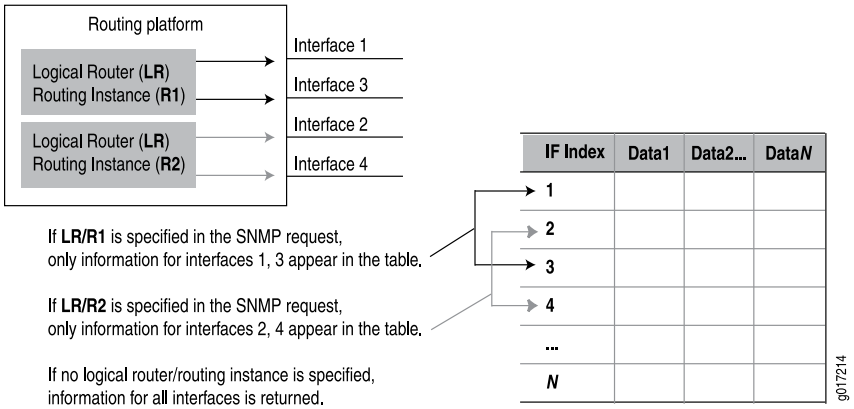
In Junos OS:

- Clients from routing instances other than the default can access MIB objects and perform SNMP operations only on the logical system networks to which they belong.
- Clients from the default routing instance can access information related to all routing instances and logical system networks.

Before Junos OS Release 8.4, only the SNMP manager in the default routing instance (**inet.0**) had access to the MIB objects

With the increase in virtual private network (VPN) service offerings, this feature is useful particularly for service providers who need to obtain SNMP data for specific routing instances (see Figure 1 on page 16). Service providers can use this information for their own management needs or export the data for use by their customers.

Figure 1: SNMP Data for Routing Instances



If no routing instance is specified in the request, the SNMP agent operates as before:

- For nonrouting table objects, all instances are exposed.
- For routing table objects, only those associated with the default routing instance are exposed.

NOTE: The actual protocol data units (PDUs) are still exchanged over the default (inet.0) routing instance, but the data contents returned are dictated by the routing instance specified in the request PDUs.

Related Documentation

- Support Classes for MIB Objects on page 17
- Trap Support for Routing Instances on page 18
- Identifying a Routing Instance on page 151
- Enabling SNMP Access over Routing Instances on page 152
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152
- Configuring Access Lists for SNMP Access over Routing Instances on page 155

Support Classes for MIB Objects

When a routing instance is specified, all routing-related MIB objects return data maintained by the routing instance in the request. For all other MIB objects, the data returned is segregated according to that routing instance. For example, only those interfaces assigned to that routing instance (for example, the logical interfaces [**ifl**s] as well as their corresponding physical interfaces [**ifd**s]) are exposed by the SNMP agent. Similarly, objects with an unambiguous attachment to an interface (for example, **addresses**) are segregated as well.

For those objects where the attachment is ambiguous (for example, objects in **sysApplMIB**), no segregation is done and all instances are visible in all cases.

Another category of objects is visible only when no logical system is specified (only within the default logical system) regardless of the routing instance within the default logical system. Objects in this category are Chassis MIB objects, objects in the SNMP group, RMON alarm, event and log groups, Ping MIB objects, configuration management objects, and V3 objects.

In summary, to support routing instances, MIB objects fall into one of the following categories:

- Class 1—Data is segregated according to the routing instance in the request. This is the most granular of the segregation classes.
- Class 2—Data is segregated according to the logical system specified in the request. The same data is returned for all routing instances that belong to a particular logical system. Typically, this applies to routing table objects where it is difficult to extract routing instance information or where routing instances do not apply.
- Class 3—Data is exposed only for the default logical system. The same set of data is returned for all routing instances that belong to the default logical system. If you specify another logical system (not the default), no data is returned. Typically this class applies

to objects implemented in subagents that do not monitor logical system changes and register their objects using only the default context (for example, Chassis MIB objects).

- Class 4—Data is not segregated by routing instance. The same data is returned for all routing instances. Typically, this applies to objects implemented in subagents that monitor logical system changes and register or deregister all their objects for each logical system change. Objects whose values cannot be segregated by routing instance fall into this class.

See "MIB Support Details" on page 67 for a list of the objects associated with each class.

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Trap Support for Routing Instances on page 18

Trap Support for Routing Instances

You can restrict the trap receivers from receiving traps that are not related to the logical system networks to which they belong. To do this, include the **logical-system-trap-filter** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
logical-system-trap-filter;
```

If the **logical-system-trap-filter** statement is not included in the SNMP configuration, all traps are forwarded to the configured routing instance destinations. However, even when this statement is configured, the trap receiver associated with the default routing instance will receive all SNMP traps.

When configured under the trap-group object, all v1 and v2c traps that apply to routing instances (or interfaces belonging to a routing instance) have the routing instance name encoded in the community string. The encoding is identical to that used in request PDUs.

For traps configured under the v3 framework, the routing instance name is carried in the context field when the v3 message processing model has been configured. For other message processing models (v1 or v2c), the routing instance name is not carried in the trap message header (and not encoded in the community string).

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Support Classes for MIB Objects on page 17
- MIB Support Details on page 67

CHAPTER 3

SNMP MIB Support

Standard SNMP MIBs Supported by Junos OS

Table 4 on page 19 contains the list of standard SNMP MIBs and RFCs that are supported on various devices running Junos OS. RFCs can be found at http://www.ietf.org .

NOTE: In this table, a value of 1 in any of the platform columns (ACX, M, T, J, MX, EX, and SRX) denotes that the corresponding MIB is supported on that particular platform, and a value of 0 denotes that the MIB is not supported on the platform.

Table 4: Standard MIBs Supported on Devices Running Junos OS

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
IEEE 802.1ab section 12.1, *Link Layer Discovery Protocol (LLDP) MIB* EX Series implementation of LLDP MIB supports both IPv4 and IPv6 configuration. For more information about LLDP MIB objects supported on EX Series devices, see *LLDP Standard MIB Objects Supported on EX Series Devices*.	0	0	0	0	1	1	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
IEEE, 802.3ad, *Aggregation of Multiple Link Segments* Supported tables and objects: • **dot3adAggPortTable**, **dot3adAggPortListTable**, **dot3adAggTable**, and **dot3adAggPortStatsTable** NOTE: EX Series switches do not support the **dot3adAggPortTable** and **dot3adAggPortStatsTable**. • **dot3adAggPortDebugTable** (only **dot3adAggPortDebugRxState**, **dot3adAggPortDebugMuxState**, **dot3adAggPortDebugActorSyncTransitionCount**, **dot3adAggPortDebugPartnerSyncTransitionCount**, **dot3adAggPortDebugActorChangeCount**, and **dot3adAggPortDebugPartnerChangeCount**) NOTE: EX Series switches do not support the **dot3adAggPortDebugTable**. • **dot3adTablesLastChanged** NOTE: Gigabit Ethernet interfaces on J Series Services Routers do not support the 802.3ad MIB.	0	1	1	1	1	1	1	1	1
RFC 1155, *Structure and Identification of Management Information for TCP/IP-based Internets*	1	1	1	1	1	1	1	1	1
RFC 1157, *A Simple Network Management Protocol (SNMP)*	1	1	1	1	1	1	1	1	1
RFC 1195, *Use of OSI IS-IS for Routing in TCP/IP and Dual Environments* (only the objects **isisSystem**, **isisMANAreaAddr**, **isisAreaAddr**, **isisSysProtSupp**, **isisSummAddr**, **isisCirc**, **isisCircLevel**, **isisPacketCount**, **isisISAdj**, **isisISAdjAreaAddr**, **isisAdjIPAddr**, **isisISAdjProtSupp**, **isisRa**, and **isisIPRA** are supported)	1	1	1	1	1	1	1	1	1
RFC 1212, *Concise MIB Definitions*	1	1	1	1	1	1	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End

RFC 1213, *Management Information Base for Network Management of TCP/IP-Based Internets: MIB-II*. Junos OS supports the following areas:
• MIB II and its SNMP version 2 derivatives, including:
  ◦ Statistics counters
  ◦ IP, except for **ipRouteTable**, which has been replaced by **ipCidrRouteTable** (RFC 2096, *IP Forwarding Table MIB*)
  ◦ SNMP management
  ◦ Interface management
• SNMPv1 **Get**, **GetNext** requests, and version 2 **GetBulk** request
• Junos OS-specific secured access list
• Master configuration keywords
• Reconfigurations upon SIGHUP
	1	1	1	1	1	1	0	0	1

RFC 1215, *A Convention for Defining Traps for use with the SNMP* (only MIB II SNMP version 1 traps and version 2 notifications)	1	1	1	1	1	1	0	0	1
RFC 1406, *Definitions of Managed Objects for the DS1 and E1 Interface Types* (T1 MIB is supported)	1	1	1	1	0	0	1	0	0
RFC 1407, *Definitions of Managed Objects for the DS3/E3 Interface Type* (T3 MIB is supported)	0	1	1	1	0	0	0	0	0
RFC 1471, *Definitions of Managed Objects for the Link Control Protocol of the Point-to-Point Protocol* (only pppLink group is supported. The pppLink group consists of the **pppLcp1** object and the tables **pppLinkStatustable** and **pppLinkConfigTable**).	0	1	0	0	1	0	0	0	0
RFC 1657, *Definitions of Managed Objects for the Fourth Version of the Border Gateway Protocol (BGP-4) using SMIv2*	1	1	1	1	1	1	0	0	0
RFC 1695, *Definitions of Managed Objects for ATM Management Version 8.0 Using SMIv2*	1	1	1	1	0	0	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 1850, *OSPF Version 2 Management Information Base* (except for the **ospfOriginateNewLsas** and **ospfRxNewLsas** objects, the Host Table, and the traps **ospfOriginateLSA**, **ospfLsdbOverflow**, and **ospfLsdbApproachingOverflow**)	1	1	1	1	1	1	1	0	0
RFC 1901, *Introduction to Community-based SNMPv2*	1	1	1	1	1	1	1	1	1
RFC 2011, *SNMPv2 Management Information Base for the Internet Protocol Using SMIv2*	1	1	1	1	1	1	0	0	0
RFC 2012, *SNMPv2 Management Information Base for the Transmission Control Protocol Using SMIv2*	1	1	1	1	1	1	1	0	1
RFC 2013, *SNMPv2 Management Information Base for the User Datagram Protocol Using SMIv2*	1	1	1	1	1	1	1	0	1
RFC 2024, *Definitions of Managed Objects for Data Link Switching Using SMIv2* (except for the **dlswInterface** and **dlswSdlc** object groups; the **dlswDirLocateMacTable**, **dlswDirNBTable**, and **dlswDirLocateNBTable** tables; the **dlswCircuitDiscReasonLocal** and **dlswCircuitDiscReasonRemote** tabular objects; and the **dlswDirMacCacheNextIndex** and **dlswDirNBCacheNextIndex** scalar objects; read-only access)	0	1	1	1	1	0	0	0	0
RFC 2096, *IP Forwarding Table MIB* (The **ipCidrRouteTable** has been extended to include the tunnel name when the next hop is through an RSVP-signaled LSP.) NOTE: RFC 2096 has been replaced by RFC 4292. However, Junos OS currently supports both RFC 2096 and RFC 4292.	1	1	1	1	1	1	0	0	1
RFC 2115, *Management Information Base for Frame Relay DTEs Using SMIv2* (**frDlcmiTable** only; **frCircuitTable** and **frErrTable** are not supported)	0	1	1	1	1	0	1	0	0
RFC 2233, *The Interfaces Group MIB Using SMIv2* NOTE: RFC 2233 has been replaced by RFC 2863, IF MIB. However, Junos OS supports both RFC 2233 and RFC 2863.	1	1	1	1	1	1	1	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 2287, *Definitions of System-Level Managed Objects for Applications* (only the objects **sysApplInstallPkgTable**, **sysApplInstallElmtTable**, **sysApplElmtRunTable**, and **sysApplMapTable**)	1	1	1	1	1	1	1	0	1
RFC 2465, *Management Information Base for IP Version 6: Textual Conventions and General Group* (except for IPv6 interface statistics)	1	1	1	1	1	0	1	0	0
RFC 2495, *Definitions of Managed Objects for the DS1, E1, DS2, and E2 Interface Types* (except for **dsx1FarEndConfigTable**, **dsx1FarEndCurrentTable**, **dsx1FarEndIntervalTable**, **dsx1FarEndTotalTable**, and **dsx1FracTable**)	1	1	1	1	0	0	1	0	0
RFC 2515, *Definitions of Managed Objects for ATM Management* (except **atmVpCrossConnectTable**, **atmVcCrossConnectTable**, and **aal5VccTable**)	1	1	1	1	0	0	0	0	0
RFC 2570, *Introduction to Version 3 of the Internet-standard Network Management Framework*	1	1	1	1	1	1	0	0	1
RFC 2571, *An Architecture for Describing SNMP Management Frameworks* (read-only access) NOTE: RFC 2571 has been replaced by RFC 3411. However, Junos OS supports both RFC 2571 and RFC 3411.	1	1	1	1	1	1	1	0	1
RFC 2572, *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)* (read-only access) NOTE: RFC 2572 has been replaced by RFC 3412. However, Junos OS supports both RFC 2572 and RFC 3412.	1	1	1	1	1	1	1	0	1
RFC 2576, *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework* NOTE: RFC 2576 has been replaced by RFC 3584. However, Junos OS supports both RFC 2576 and RFC 3584.	1	1	1	1	1	1	1	0	1
RFC 2578, *Structure of Management Information Version 2 (SMIv2)*	1	1	1	1	1	1	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 2579, *Textual Conventions for SMIv2*	1	1	1	1	1	1	0	0	1
RFC 2580, *Conformance Statements for SMIv2*	1	1	1	1	1	1	0	0	1
RFC 2662, *Definitions of Managed Objects for ADSL Lines* (J Series Services Routers. All MIB tables, objects, and traps are applicable for the ADSL ATU-R agent.)	0	1	1	1	1	0	1	0	0
RFC 2665, *Definitions of Managed Objects for the Ethernet-like Interface Types* NOTE: For M, T and MX Series, the SNMP counters do not count the Ethernet header and frame check sequence (FCS). Therefore, the following four OIDs are not supported: • ifInOctets • ifOutOctets • ifHCInOctets • ifHCOutOctets However, the EX switches adhere to RFC 2665.	1	1	1	1	1	1	1	0	1
RFC 2787, *Definitions of Managed Objects for the Virtual Router Redundancy Protocol* (except row creation, the **Set** operation, and the object **vrrpStatsPacketLengthErrors**)	1	1	1	1	1	1	1	0	1
RFC 2790, *Host Resources MIB* • Only the **hrStorageTable**. The file systems **/**, **/config**, **/var**, and **/tmp** always return the same index number. When SNMP restarts, the index numbers for the remaining file systems might change. • Only the objects of the **hrSystem** and **hrSWInstalled** groups.	1	1	1	1	1	1	1	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 2819, *Remote Network Monitoring Management Information Base* • **etherStatsTable** (for Ethernet interfaces only),**alarmTable**, **eventTable**, and **logTable** are supported on all devices running Junos OS. • **historyControlTable** and **etherHistoryTable** (except **etherHistoryUtilization** object) are supported only on EX Series switches.	1	1	1	1	1	1	1	0	1
RFC 2863, *The Interfaces Group MIB* NOTE: RFC 2863 replaces RFC 2233. However, Junos OS supports both RFC 2233 and RFC 2863.	1	1	1	1	1	1	0	0	1
RFC 2864, *The Inverted Stack Table Extension to the Interfaces Group MIB*	0	1	1	1	1	0	0	0	1
RFC 2922, *The Physical Topology (PTOPO) MIB* Supported objects: **ptopoConnDiscAlgorithm, ptopoConnAgentNetAddrType, ptopoConnAgentNetAddr, ptopoConnMultiMacSASeen, ptopoConnMultiNetSASeen, ptopoConnIsStatic, ptopoConnLastVerifyTime, ptopoConnRowStatus**	0	0	0	0	0	1	1	0	1
RFC 2925, *Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations* (only the objects **pingCtlTable**, **pingResultsTable**, **pingProbeHistoryTable**, **pingMaxConcurrentRequests**, **traceRouteCtlTable**, **traceRouteResultsTable**, **traceRouteProbeHistoryTable**, and **traceRouteHopsTable**)	1	1	1	1	1	1	1	0	1
RFC 2932, *IPv4 Multicast Routing MIB*	1	1	1	1	1	1	1	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 2934, *Protocol Independent Multicast MIB for IPv4* NOTE: In Junos OS, RFC 2934 is implemented based on a draft version, *pimmib.mib*, of the now standard RFC. Support for the **pimNeighborLoss** trap was added in Release 11.4.	1	1	1	1	1	1	1	0	0
RFC 2981, *Event MIB*	1	1	1	1	1	0	0	0	0
RFC 3014, *Notification Log MIB*	1	1	1	1	1	0	0	0	0
RFC 3019, *IP Version 6 Management Information Base for The Multicast Listener Discovery Protocol*	0	1	1	1	1	0	0	0	1
RFC 3410 *Introduction and Applicability Statements for Internet-Standard Management Framework*	1	1	1	1	1	1	0	0	1
RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks* NOTE: RFC 3411 replaces RFC 2571. However, Junos OS supports both RFC 3411 and RFC 2571.	1	1	1	1	1	1	0	0	1
RFC 3412, *Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)* NOTE: RFC 3412 replaces RFC 2572. However, Junos OS supports both RFC 3412 and RFC 2572.	1	1	1	1	1	1	0	0	1
RFC 3413, *Simple Network Management Protocol (SNMP) Applications* (except for the Proxy MIB)	1	1	1	1	1	1	1	0	1
RFC 3414, *User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*	1	1	1	1	1	1	0	0	1
RFC 3415, *View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*	1	1	1	1	1	1	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 3416, *Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)* NOTE: RFC 3416 replaces RFC 1905, which was supported in earlier versions of Junos OS.	1	1	1	1	1	1	0	0	1
RFC 3417, *Transport Mappings for the Simple Network Management Protocol (SNMP)*	1	1	1	1	1	1	1	0	1
RFC 3418, *Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)* NOTE: RFC 3418 replaces RFC 1907, which was supported in earlier versions of Junos OS.	1	1	1	1	1	1	0	0	1
RFC 3498, *Definitions of Managed Objects for Synchronous Optical Network (SONET) Linear Automatic Protection Switching (APS) Architectures* (implemented under the Juniper Networks enterprise branch [**jnxExperiment**])	0	1	1	1	0	0	0	0	0
RFC 3584 *Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*	1	1	1	1	1	1	0	0	1
RFC 3591 *Managed Objects for the Optical Interface Type* **optIfOTMnTable** (except **optIfOTMnOpticalReach**, **optIfOTMnInterfaceType**, and **optIfOTMnOrder**), **optIfOChConfigTable** (except **optIfOChDirectionality** and **optIfOChCurrentStatus**), **optIfOTUkConfigTable** (except **optIfOTUkTraceIdentifierAccepted**, **optIfOTUkTIMDetMode**, **optIfOTUkTIMActEnabled**, **optIfOTUkTraceIdentifierTransmitted**, **optIfOTUkDEGThr**, **optIfOTUkDEGM**, **optIfOTUkSinkAdaptActive**, and **optIfOTUkSourceAdaptActive**), and **optIfODUkConfigTable** (except **optIfODUkPositionSeqCurrentSize** and **optIfODUkTtpPresent**)	0	1	1	1	0	0	0	0	0
RFC 3592, *Definitions of Managed Objects for the Synchronous Optical Network/Synchronous Digital Hierarchy (SONET/SDH) Interface Type*	0	1	1	1	1	0	0	0	0
RFC 3621, *Power Ethernet MIB*	0	0	0	0	0	1	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 3637, *Definitions of Managed Objects for the Ethernet WAN Interface Sublayer* (except **etherWisDeviceTable**, **etherWisSectionCurrentTable**, and **etherWisFarEndPathCurrentTable**)	0	1	1	1	1	0	0	0	0
RFC 3811, *Definitions of Textual Conventions (TCs) for Multiprotocol Label Switching (MPLS) Management*	1	1	1	1	1	0	1	0	0
RFC 3812, *Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Management Information Base (MIB)* (read only access) • MPLS tunnels as interfaces are not supported. • The following objects in the **TunnelResource** table are not supported: **mplsTunnelResourceMeanRate**, **mplsTunnelResourceMaxBurstSize**, **mplsTunnelResourceMeanBurstSize**, **mplsTunnelResourceExBurstSize**, **mplsTunnelResourceWeight**. • **mplsTunnelPerfTable** and **mplsTunnelCRLDPResTable** are not supported. • **mplsTunnelCHopTable** is supported on ingress routers only. NOTE: The branch used by the proprietary LDP MIB (**ldpmib.mib**) conflicts with RFC 3812. **ldpmib.mib** has been deprecated and replaced by **jnx-mpls-ldp.mib**.	1	1	1	1	1	0	0	0	0
RFC 3813, *Multiprotocol Label Switching (MPLS) Label Switching Router (LSR) Management Information Base (MIB)* (read-only access). **mplsInterfacePerfTable**, **mplsInSegmentPerfTable**, **mplsOutSegmentPerfTable**, **mplsInSegmentMapTable**, **mplsXCUp**, and **mplsXCDown** are not supported.	1	1	1	1	1	0	1	0	0
RFC 3826, *The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model*	1	1	1	1	1	1	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 3896, *Definitions of Managed Objects for the DS3/E3 Interface Type* (except **dsx3FarEndConfigTable**, **dsx3FarEndCurrentTable**, **dsx3FarEndIntervalTable**, **dsx3FarEndTotalTable**, and **dsx3FracTable**)	0	1	1	1	0	0	0	0	0
RFC 4087, *IP Tunnel MIB*—Describes MIB objects in the following tables for managing tunnels of any type over IPv4 and IPv6 networks: • **tunnelIfTable**—Provides information about the tunnels known to a router. • **tunnelInetConfigTable**—Assists dynamic creation of tunnels and provides mapping from end-point addresses to the current interface index value. NOTE: Junos OS supports **MAX-ACCESS** of read-only for all the MIB objects in **tunnelIfTable** and **tunnelInetConfigTable** tables.	0	1	1	0	1	0	0	0	0
RFC 4188, *Definitions of Managed Objects for Bridges*—Supports 802.1D STP(1998). Supports only the following subtrees and objects: • **dot1dStp** subtree is supported on MX Series 3D Universal Edge Routers . • **dot1dTpFdbAddress**, **dot1dTpFdbPort**, and **dot1dTpFdbStatus** objects from the **dot1dTpFdbTable** of the **dot1dTp** subtree are supported on EX Series Ethernet Switches. NOTE: **dot1dTpLearnedEntryDiscards** and **dot1dTpAgingTime** objects are supported on M and T Series routers.	0	0	0	0	1	1	0	0	0
RFC 4273, *Definitions of Managed Objects for BGP-4* (only **jnxBgpM2PrefixInPrefixes**, **jnxBgpM2PrefixInPrefixesAccepted**, and **jnxBgpM2PrefixInPrefixesRejected** objects)	1	1	1	1	1	1	0	0	1
RFC 4273, *Definitions of Managed Objects for BGP-4* (only **jnxBgpM2PrefixInPrefixes**, **jnxBgpM2PrefixInPrefixesAccepted**, and **jnxBgpM2PrefixInPrefixesRejected** objects)	1	1	1	1	1	1	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 4292, *IP Forwarding MIB*— Describes a table and MIB objects for forwarding IP packets that are version independent: • **inetCidrRouteTable**—Provides the ability to display IP version-independent multipath CIDR routes and obsoletes the **ipCidrRouteTable** object. • **inetCidrRouteNumber**—Indicates the number of current routes and obsoletes the **ipCidrRouteNumber** object. • **inetCidrRouteDiscards**—Counts the number of valid routes that are discarded from **inetCidrRouteTable** and obsoletes the **ipCidrRouteDiscards** object. NOTE: Junos OS currently supports these MIB objects that will be deprecated in future releases: **ipCidrRouteTable**, **ipCidrRouteNumber**, and **ipCidrRouteDiscards**.	1	1	1	1	1	1	0	0	0
RFC 4293, *Management Information Base for the Internet Protocol (IP)* — Supports only the mandatory groups. For detailed information, see Standard IPv4/IPv6 MIBs.	0	0	0	0	1	1	0	0	0
RFC 4318, *Definitions of Managed Objects for Bridges with Rapid Spanning Tree Protocol*—Supports 802.1w and 802.1t extensions for RSTP.	0	1	1	1	1	1	0	0	0
RFC 4363b, *Q-Bridge VLAN MIB*	0	0	0	0	1	1	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 4382 *MPLS/BGP Layer 3 Virtual Private Network (VPN) MIB* The Junos OS support for RFC 4382 includes the following scalar objects and tables: • **mplsL3VpnActiveVrfs** • **mplsL3VpnConfiguredVrfs** • **mplsL3VpnConnectedInterfaces** • **mplsL3VpnVrfConfMidRteThresh** • **mplsL3VpnVrfConfHighRteThresh** • **mplsL3VpnIfConfRowStatus** • **mplsL3VpnIllLblRcvThrsh** • **mplsL3VpnNotificationEnable** • **mplsL3VpnVrfConfMaxPossRts** • **mplsL3VpnVrfConfRteMxThrshTime** • **mplsL3VpnVrfOperStatus** • **mplsL3VpnVrfPerfCurrNumRoutes** • **mplsL3VpnVrfPerfTable** • **mplsL3VpnVrfRteTable** • **mplsVpnVrfRTTable** • **mplsL3VpnVrfTable** NOTE: The **mplsL3VpnIfConfTable** has not been implemented in the MPLS/BGP Layer 3 Virtual Private Network (VPN) MIB, because of limited utility and difficulty in representing the **DistProtocol** bit accurately.	0	1	1	1	1	1	0	0	0
RFC 4444, *IS-IS MIB*	1	1	1	1	1	1	1	0	0
RFC 4668, *RADIUS Accounting Client Management Information Base (MIB) for IPv6* (read-only access)	0	0	0	0	1	0	0	0	0
RFC 4670, *RADIUS Accounting Client Management Information Base (MIB)* (read-only access)	0	0	0	0	1	0	0	0	0
RFC 4801, *Definitions of Textual Conventions for Generalized Multiprotocol Label Switching (GMPLS) Management Information Base (MIB)* (read-only access)	0	1	1	1	1	0	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 4802, *Generalized Multiprotocol Label Switching (GMPLS) Traffic Engineering (TE) Management Information Base (MIB)* (read-only access). **gmplsTunnelReversePerfTable**, **gmplsTeScalars**, **gmplsTunnelTable**, **gmplsTunnelARHopTable**, **gmplsTunnelCHopTable**, and **gmplsTunnelErrorTable** are not supported.)	0	1	1	1	1	0	0	0	0
RFC 4803, *Generalized Multiprotocol Label Switching (GMPLS) Label Switching Router (LSR) Management Information Base (MIB)* (read-only access). **gmplsLabelTable** and **gmplsOutsegmentTable** are not supported. NOTE: The tables in GMPLS TE (RFC 4802) and LSR (RFC 4803) MIBs are extensions of the corresponding tables from the MPLS TE (RFC 3812) and LSR (RFC 3813) MIBs and use the same index as the MPLS MIB tables.	0	1	1	1	1	0	0	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
RFC 5643, *Management Information Base for OSPFv3* NOTE: Junos OS support for this MIB is read-only. Junos OS does not support the following tables and objects defined in this MIB. • **ospfv3HostTable** • **ospfv3CfgNbrTable** • **ospfv3ExitOverflowInterval** • **ospfv3ReferenceBandwidth** • **ospfv3RestartSupport** • **ospfv3RestartInterval** • **ospfv3RestartStrictLsaChecking** • **ospfv3RestartStatus** • **ospfv3RestartAge** • **ospfv3RestartExitReason** • **ospfv3NotificationEnable** • **ospfv3StubRouterSupport** • **ospfv3StubRouterAdvertisement** • **ospfv3DiscontinuityTime** • **ospfv3RestartTime** • **ospfv3AreaNssaTranslatorRole** • **ospfv3AreaNssaTranslatorState** • **ospfv3AreaNssaTranslatorStabInterval** • **ospfv3AreaNssaTranslatorEvents** • **ospfv3AreaTEEnabled** • **ospfv3IfMetricValue** • **ospfv3IfDemandNbrProbe**	0	1	1	1	1	0	0	0	1
RFC 6527, *Definitions of Managed Objects for the Virtual Router Redundancy Protocol Version 3 (VRRPv3)* (except row creation, the **Set** operation, and the objects **vrrpv3StatisticsRowDiscontinuityTime** and **vrrpv3StatisticsPacketLengthErrors**)	1	0	0	1	0	0	0	0	0
Internet Assigned Numbers Authority, *IANAiftype Textual Convention MIB* (referenced by RFC 2233, available at http://www.iana.org/assignments/ianaiftype-mib)	1	1	1	1	1	1	1	0	0

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
Internet draft draft-ietf-atommib-sonetaps-mib-10.txt, *Definitions of Managed Objects for SONET Linear APS Architectures* (as defined under the Juniper Networks enterprise branch [**jnxExperiment**] only)	0	1	1	1	1	0	0	0	0
Internet draft draft-ieft-bfd-mib-02.txt, *Bidirectional Forwarding Detection Management Information Base* (Represented by **mib-jnx-bfd-exp.txt** and implemented under the Juniper Networks enterprise branch [**jnxExperiment**]. Read only. Includes **bfdSessUp** and **bfdSessDown** traps. Does not support **bfdSessPerfTable** and **bfdSessMapTable**.)	1	1	1	1	1	1	0	0	1
Internet draft draft-ietf-idmr-igmp-mib-13.txt, *Internet Group Management Protocol (IGMP) MIB*	0	1	1	1	1	1	0	0	1
Internet draft draft-reeder-snmpv3-usm-3desede-00.txt, *Extension to the User-Based Security Model (USM) to Support Triple-DES EDE in 'Outside' CBC Mode*	1	1	1	1	1	1	0	0	1
Internet draft draft-ietf-isis-wg-mib-07.txt, *Management Information Base for IS-IS* (only **isisISAdjTable**, **isisISAdjAreaAddrTable**, **isisISAdjIPAddrTable**, and **isisISAdjProtSuppTable)** NOTE: Replaced with RFC 4444, *IS-IS MIB* in Junos OS Release 11.3 and later.	1	1	1	1	1	1	1	0	0
Internet draft draft-ietf-ppvpn-mpls-vpn-mib-04.txt, *MPLS/BGP Virtual Private Network Management Information Base Using SMIv2* (only **mplsVpnScalars**, **mplsVpnVrfTable, mplsVpnPerTable**, and **mplsVpnVrfRouteTargetTable)**	0	1	1	1	1	0	0	0	0
Internet draft draft-ietf-ospf-ospfv3-mib-11.txt, *Management Information Base for OSPFv3* (Represented by **mib-jnx-ospfv3mib.txt** and implemented under the Juniper Networks enterprise branch **{jnxExperiment}**. Support for **ospfv3NbrTable** only. Read only. Object names are prefixed by **jnx**. For example, **jnxOspfv3NbrTable**, **jnxOspfv3NbrAddressType**, and **jnxOspfv3NbrPriority**.)	0	1	1	1	1	0	0	0	1

Table 4: Standard MIBs Supported on Devices Running Junos OS *(continued)*

MIB/RFC	Platforms								
							SRX		
	ACX	M	T	J	MX	EX	Low-End	Mid-Range	High-End
Internet draft draft-ietf-idmr-pim-mib-09.txt, *Protocol Independent Multicast (PIM) MIB*	1	1	1	1	1	1	0	0	1
ESO Consortium MIB, which can be found at http://www.snmp.com/eso/ NOTE: The ESO Consortium MIB has been replaced by RFC 3826.	1	1	1	1	1	1	1	0	0
Internet Draft P2MP MPLS-TE MIB (draft-ietf-mpls-p2mp-te-mib-09.txt) (read-only access) (except **mplsTeP2mpTunnelBranchPerfTable**).	1	1	1	1	1	0	0	0	0

Related Documentation

- Juniper Networks Enterprise-Specific MIBs on page 35
- Loading MIB Files to a Network Management System on page 297

Juniper Networks Enterprise-Specific MIBs

Junos OS supports the following enterprise-specific MIBs:

- AAA Objects MIB—Provides support for monitoring user authentication, authorization, and accounting through the RADIUS, LDAP, SecurID, and local authentication servers. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.Juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-user--aaa.txt

  For more information, see *AAA Objects MIB*.

- Access Authentication Objects MIB—Provides support for monitoring firewall authentication, including data about the users trying to access firewall-protected resources and the firewall authentication service itself. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.Juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-auth.txt.

  For more information, see *Access Authentication Objects MIB*.

- Alarm MIB—Provides support for alarms from the router. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis-alarm.txt

  For more information, see *Alarm MIB*.

- Analyzer MIB—Contains analyzer and remote analyzer data related to port mirroring on the EX Series Ethernet Switches. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-analyzer.txt

  For more information, see *Analyzer MIB*.

- Antivirus Objects MIB—Provides information about the antivirus engine, antivirus scans, and antivirus scan-related traps. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-utm-av.txt

  For more information, see *Antivirus Objects MIB*.

- ATM Class-of-Service MIB—Provides support for monitoring Asynchronous Transfer Mode, version 2 (ATM2) virtual circuit (VC) class-of-service (CoS) configurations. It also provides CoS queue statistics for all VCs that have CoS configured. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-atm-cos.txt

  For more information, see *ATM Class-of-Service MIB*.

- ATM MIB—Provides support for ATM interfaces and virtual connections. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-atm.txt.

  For more information, see *ATM MIB*.

- BGP4 V2 MIB—Contains objects used to monitor BGP peer-received prefix counters. It is based upon similar objects in the MIB documented in Internet draft draft-ietf-idr-bgp4-mibv2-03.txt, *Definitions of Managed Objects for the Fourth Version of BGP (BGP-4), Second Version*. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-bgpmib2.txt

  For more information, see *BGP4 V2 MIB*.

- Bidirectional Forwarding Detection MIB—Provides support for monitoring Bidirectional Forwarding Detection (BFD) sessions. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-bfd.txt

  For more information, see *Bidirectional Forwarding Detection MIB*.

- Chassis Definitions for Router Model MIB—Contains the object identifiers (OIDs) that are used by the Chassis MIB to identify platform and chassis components. The Chassis MIB provides information that changes often, whereas the Chassis Definitions for Router Model MIB provides information that changes less often. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chas-defines.txt

  For more information, see *Chassis MIBs*.

- Chassis Forwarding MIB—Enables J Series Services Routers to fully support the Junos OS health monitor. This MIB extends the scope of health monitoring to include Junos forwarding process (**fwdd**) components. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis-fwdd.txt

  For more information, see *Chassis Forwarding MIB*.

- Chassis MIB—Provides support for environmental monitoring (power supply state, board voltages, fans, temperatures, and air flow) and inventory support for the chassis, System Control Board (SCB), System and Switch Board (SSB), Switching and Forwarding Module (SFM), Flexible PIC Concentrators (FPCs), and PICs. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis.txt

  For more information, see *Chassis MIBs*.

- Chassis Cluster MIB—Provides information about objects that are used whenever the state of the control link interfaces or fabric link interfaces changes (up to down or down to up) in a chassis cluster deployment. For a downloadable version of this MIB, see http://juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jsrpd.txt.

  For more information, see *Chassis Cluster MIB* .

- Class-of-Service MIB—Provides support for monitoring interface output queue statistics per interface and per forwarding class. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-cos.txt

  For more information, see *Class-of-Service MIB*.

- Configuration Management MIB—Provides notification for configuration changes as SNMP traps. Each trap contains the time at which the configuration change was committed, the name of the user who made the change, and the method by which the change was made. A history of the last 32 configuration changes is kept in **jnxCmChgEventTable**. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-cfgmgmt.txt

  For more information, see *Configuration Management MIB*.

- Destination Class Usage MIB—Provides support for monitoring packet counts based on the ingress and egress points for traffic transiting your networks. Ingress points are identified by the input interface. Egress points are identified by destination prefixes grouped into one or more sets, known as destination classes. One counter is managed per interface per destination class, up to a maximum of 16 counters per interface. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dcu.txt

  For more information, see *Destination Class Usage MIB*.

- DHCP Objects MIB— Provides SNMP support (get and trap) for DHCP local server and relay configurations. It also provides support for bindings and leases tables, and for statistics. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jdhcp.txt

  For more information, see *DHCP MIB*.

- DHCPv6 MIB—Provides SNMP support (get and trap) for DHCPv6 local server and relay configurations. It also provides support for bindings and leases tables, and for statistics. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jdhcpv6.txt

  For more information, see *DHCPv6 MIB*.

- Digital Optical Monitoring MIB—Provides support for the **SNMP Get** request for statistics and **SNMP Trap** notifications for alarms. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dom.txt

  For more information, see *Digital Optical Monitoring MIB*.

- DNS Objects MIB—Provides support for monitoring DNS proxy queries, requests, responses, and failures. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-dns.txt

  For more information, see *DNS Objects MIB*.

- Dynamic Flow Capture MIB—Provides support for monitoring the operational status of dynamic flow capture (DFC) PICs. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dfc.txt

  For more information, see *Dynamic Flow Capture MIB*.

- Ethernet MAC MIB—Monitors media access control (MAC) statistics on Gigabit Ethernet intelligent queuing (IQ) interfaces. It collects MAC statistics; for example, **inoctets**, **inframes**, **outoctets**, and **outframes** on each source MAC address and virtual LAN (VLAN) ID for each Ethernet port. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mac.txt

  For more information, see *Ethernet MAC MIB*.

- Event MIB—Defines a generic trap that can be generated using an op script or event policy. This MIB provides the ability to specify a system log string and raise a trap if that system log string is found. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-event.txt

  For more information, see *Event MIB*.

- Experimental MIB—Contains object identifiers for experimental MIBs. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-exp.txt

  For more information, see *jnxExperiment*.

- Firewall MIB—Provides support for monitoring firewall filter counters. Routers must have the Internet Processor II ASIC to perform firewall monitoring. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-firewall.txt

  For more information, see *Firewall MIB*.

- Flow Collection Services MIB—Provides statistics on files, records, memory, FTP, and error states of a monitoring services interface. It also provides SNMP traps for unavailable destinations, unsuccessful file transfers, flow overloading, and memory overloading. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-coll.txt

  For more information, see *Flow Collection Services MIB*.

- Host Resources MIB—Extends the **hrStorageTable** object, providing a measure of the usage of each file system on the router in percentage format. Previously, the objects

in the **hrStorageTable** measured the usage in allocation units—**hrStorageUsed** and **hrStorageAllocationUnits**—only. Using the percentage measurement, you can more easily monitor and apply thresholds on usage. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-hostresources.txt

For more information, see *Host Resources MIB*.

- IDP Objects MIB—Provides support for monitoring SNMP IDP queries, requests, responses, and failures. This MIB defines the key monitoring and threshold crossing trap support, IDP database update status and trap support, attack-related monitoring and trap support for SRX100, SRX210, SRX220, SRX240, SRX550, and SRX650 Services Gateways. This MIB models IDP attributes specific to the appropriate Juniper Networks implementation. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-hostresources.txt

  For more information, see *IDP MIB*.

- Interface MIB—Extends the standard **ifTable** (RFC 2863) with additional statistics and Juniper Networks enterprise-specific chassis information. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-if-extensions.txt

  For more information, see *Interface MIB*.

- IP Forward MIB—Extends the standard IP Forwarding Table MIB (RFC 4292) to include CIDR forwarding information. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipforward.txt

  For more information, see *IP Forwarding MIB*.

- IPsec Generic Flow Monitoring Object MIB—Based on **jnx-ipsec-monitor-mib**, this MIB provides support for monitoring IPsec and IPsec VPN management objects. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipsec-flow-mon.txt

  For more information, see *IPsec Generic Flow Monitoring Object MIB*.

- IPsec Monitoring MIB—Provides operational and statistical information related to the IPsec and IKE tunnels on Juniper Networks routers. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipsec-monitor-asp.txt .

  For more information, see *IPSec Monitoring MIB*.

- IPsec VPN Objects MIB—Provides support for monitoring IPsec and IPsec VPN management objects for Juniper security product lines. This MIB is an extension of **jnx-ipsec-flow-mon.mib**. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-ipsec-vpn.txt

  For more information, see *IPsec VPN Objects MIB*.

- IPv4 MIB—Provides additional Internet Protocol version 4 (IPv4) address information, supporting the assignment of identical IPv4 addresses to separate interfaces. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipv4.txt

  For more information, see *IPv4 MIB*.

- IPv6 and ICMPv6 MIB—Provides IPv6 and Internet Control Message Protocol version 6 (ICMPv6) statistics. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipv6.txt

  For more information, see *IPv6 MIB*.

- L2ALD MIB—Contains information about the Layer 2 Address Learning Daemon (L2ALD) and related traps, such as the routing instance MAC limit trap and the interface MAC limit trap. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2ald.txt

  For more information, see *L2ALD MIB*.

- L2CP MIB—Provides information about Layer 2 Control Protocols (L2CP) based features on MX Series 3D Universal Edge Routers . Currently, Junos OS supports only the **jnxDot1dStpPortRootProtectEnabled**, **jnxDot1dStpPortRootProtectState**, and **jnxPortRootProtectStateChangeTrap** objects. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2cp-features.txt

  For more information, see *L2CP MIB*.

- L2TP MIB—Provides information about Layer 2 Transport Protocol (L2TP) tunnels and sessions. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2tp.txt

  For more information, see *L2TP MIB*.

- LDP MIB—Provides LDP statistics and defines LDP label-switched path (LSP) notifications. LDP traps support only IPv4 standards. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ldp.txt

  For more information, see *LDP MIB*.

- License MIB—Extends SNMP support to licensing information, and introduces SNMP traps that alert users when the licenses are about to expire, expire, or when the total number of users exceeds the number specified in the license. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-license.txt

  For more information, see *License MIB*.

- Logical Systems MIBs—Extend SNMP support to logical systems security profile through various MIBs defined under **jnxLsysSecurityProfile**. For a downloadable versions of the MIBs, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-lsys-securityprofile.txt

  For more information, see *Logical Systems MIB*.

- MIMSTP MIB—Provides information about MSTP instances (that is, routing instances of type Virtual Switch/Layer 2 control, also known as virtual contexts), MSTIs within the MSTP instance, and VLANs associated with the MSTI. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mimstp.txt

  For more information, see *MIMSTP MIB*.

- MPLS MIB—Provides MPLS information and defines MPLS notifications. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mpls.txt

  NOTE: **To collect information about MPLS statistics on transit routers, use the enterprise-specific RSVP MIB (mib-jnx-rsvp.txt) instead of the enterprise-specific MPLS MIB (mib-jnx-mpls.txt).**

  For more information, see *MPLS MIB*.

- MPLS LDP MIB—Contains object definitions as described in RFC 3815, *Definitions of Managed Objects for the Multiprotocol Label Switching (MPLS), Label Distribution Protocol (LDP)*. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mpls-ldp.txt

  NOTE: **Objects in the MPLS LDP MIB were supported in earlier releases of Junos OS as a proprietary LDP MIB (mib-ldpmib.txt). Because the branch used by the proprietary LDP (mib-ldpmib.txt) conflicts with RFC 3812, the proprietary LDP MIB (mib-ldpmib.txt) has been deprecated and replaced by the enterprise-specific MPLS LDP MIB (mib-jnx-mpls-ldp.txt).**

  For more information, see *MPLS LDP MIB*.

- NAT Objects MIB—Provides support for monitoring network address translation (NAT). This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-nat.txt

  For more information, see *NAT Objects MIB*.

- NAT Resources-Monitoring MIB—Provides support for monitoring NAT pools usage and NAT rules. Notifications of usage of NAT resources are also provided by this MIB. This MIB is currently supported on the Multiservices PIC and Multiservices DPC on M Series and MX Series routers only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sp-nat.txt

  For more information, see *Network Address Translation Resources–Monitoring MIB*.

- OTN Interface Management MIB—Defines objects for managing Optical Transport Network (OTN) interfaces on devices running Junos OS. For a downloadable version of the MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-otn.txt

For more information, see *OTN Interface Management MIB*.

- Packet Forwarding Engine MIB—Provides notification statistics for Packet Forwarding Engines. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pfe.txt

  For more information, see *Packet Forwarding Engine MIB*.

- Packet Mirror MIB—Enables you to capture and view packet mirroring-related information. This MIB is currently supported by Junos OS for MX Series routers only. Packet mirroring traps are an extension of the standard SNMP implementation and are only available to SNMPv3 users. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-packet-mirror.txt

  For more information, see *Packet Mirror MIB Overview*.

- PAE Extension MIB—Extends the standard IEEE802.1x PAE Extension MIB, and contains information for Static MAC Authentication. The enterprise-specific PAE Extension MIB is supported only on EX Series Ethernet Switches. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pae-extension.txt

  For more information, see *PAE Extension MIB*.

- Passive Monitoring MIB—Performs traffic flow monitoring and lawful interception of packets transiting between two routers. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pmon.txt

  For more information, see *Passive Monitoring MIB*.

- Ping MIB—Extends the standard Ping MIB control table (RFC 2925). Items in this MIB are created when entries are created in **pingCtlTable** of the Ping MIB. Each item is indexed exactly as it is in the Ping MIB. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ping.txt

  For more information, see *PING MIB*.

- Policy Objects MIB—Provides support for monitoring the security policies that control the flow of traffic from one zone to another. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-policy.txt

  For more information, see *Policy Objects MIB*.

- Power Supply Unit MIB—Enables monitoring and managing of the power supply on a device running Junos OS. This MIB is currently supported only on EX Series Ethernet Switches. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-power-supply-unit.txt

  For more information, see *Power Supply Unit MIB*.

- PPP MIB—Provides SNMP support for PPP-related information such as the type of authentication used, interface characteristics, status, and statistics. This MIB is supported on Common Edge PPP process, jpppd. This MIB is currently supported only on M Series and MX Series routers. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ppp.txt

For more information, see *PPP MIB*.

- PPPoE MIB—Provides SNMP support for PPPoE-related information such as the type of authentication used, interface characteristics, status, and statistics. This MIB is supported on Common Edge PPPoE process, jpppoed. This MIB is currently supported only on M Series and MX Series routers. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pppoe.txt

  For more information, see *PPPoE MIB*.

- Pseudowire TDM MIB—Extends the standard Pseudowire MIB, and contains information about configuration and statistics for specific pseudowire types. The enterprise-specific Pseudowire TDM MIB is the Juniper Networks implementation of the standard Managed Objects for TDM over Packet Switched Network MIB (draft-ietf-pwe3-tdm-mib-08.txt). For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pwtdm.txt

  For more information, see *Pseudowire TDM MIB*.

- Pseudowire ATM MIB—Extends the standard Pseudowire MIB, and defines objects used for managing the ATM pseudowires in Juniper products. The enterprise-specific Pseudowire ATM MIB is the Juniper Networks implementation of RFC 5605, *Managed Objects for ATM over Packet Switched Networks (PSNs)*. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pwatm.txt

  For more information, see *Pseudowire ATM MIB*.

- Real-Time Performance Monitoring MIB—Provides real-time performance-related data and enables you to access jitter measurements and calculations using SNMP. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rpm.txt

  For more information, see *Real-Time Performance Monitoring MIB*.

- Reverse-Path-Forwarding MIB—Monitors statistics for traffic that is rejected because of reverse-path-forwarding (RPF) processing. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rpf.txt

  NOTE: **The enterprise-specific RPF MIB is not supported on EX Series Ethernet Switches.**

  For more information, see *Reverse Path Forwarding MIB*.

- RMON Events and Alarms MIB—Supports the Junos OS extensions to the standard Remote Monitoring (RMON) Events and Alarms MIB (RFC 2819). The extension augments **alarmTable** with additional information about each alarm. Two new traps are also defined to indicate when problems are encountered with an alarm. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rmon.txt

  For more information, see *RMON Events and Alarms MIB*.

- RSVP MIB—Provides information about RSVP-traffic engineering sessions that correspond to MPLS LSPs on transit routers in the service provider core network. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rsvp.txt

  NOTE: **To collect information about MPLS statistics on transit routers, use the enterprise-specific RSVP MIB (mib-jnx-rsvp.txt) instead of the enterprise-specific MPLS MIB (mib-jnx-mpls.txt).**

  For more information, see *RSVP MIB*.

- Security Interface Extension Objects MIB—Provides support for the security management of interfaces. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-if-ext.txt

  For more information, see *Security Interface Extension Objects MIB*.

- Security Screening Objects MIB—Defines the MIB for the Juniper Networks Enterprise Firewall screen functionality. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-screening.txt

  For more information, see *Security Screening Objects MIB*.

- Services PIC MIB—Provides statistics for Adaptive Services (AS) PICs and defines notifications for AS PICs. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sp.txt

  For more information, see *Services PIC MIB*.

- SONET APS MIB—Monitors any SONET interface that participates in Automatic Protection Switching (APS). For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sonetaps.txt

  For more information, see *SONET APS MIB*.

- SONET/SDH Interface Management MIB—Monitors the current alarm for each SONET/SDH interface. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sonetaps.txt

  For more information, see *SONET/SDH Interface Management MIB*.

- Source Class Usage MIB—Counts packets sent to customers by performing a lookup on the IP source address and the IP destination address. The Source Class Usage (SCU) MIB makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-scu.txt

  For more information, see *Source Class Usage MIB*.

- SPU Monitoring MIB—Provides support for monitoring SPUs on SRX5600 and SRX5800 devices. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-spu-monitoring.txt

  For more information, see *SPU Monitoring Objects MIB*.

- Structure of Management Information MIB—Explains how the Juniper Networks enterprise-specific MIBs are structured. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-smi.txt

  For more information, see *Structure of Management Information MIB*.

- Structure of Management Information MIB for EX Series Ethernet Switches—Defines a MIB branch for switching-related MIB definitions for the EX Series Ethernet Switches. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ex-smi.txt

  For more information, see *EX Series SMI MIB*.

- Structure of Management Information MIB—Contains object identifiers (OIDs) for the security branch of the MIBs used in Junos OS for J Series and SRX Series devices, services, and traps. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-smi.txt

- Subscriber MIB—Provides SNMP support for subscriber-related information. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-subscriber.txt

  For more information, see *Subscriber MIB* .

- System Log MIB—Enables notification of an SNMP trap-based application when an important system log message occurs. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-syslog.txt

  For more information, see *System Log MIB*.

- Traceroute MIB—Supports the Junos OS extensions of traceroute and remote operations. Items in this MIB are created when entries are created in the **traceRouteCtlTable** of the Traceroute MIB. Each item is indexed exactly the same way as it is in the Traceroute MIB. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-traceroute.txt

  For more information, see *Traceroute MIB*.

- Utility MIB—Provides SNMP support for exposing the Junos OS data and has tables that contain information about each type of data, such as integer and string. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-util.txt

  For more information, see *Utility MIB*.

- Virtual Chassis MIB—Contains information about the virtual chassis on the EX Series Ethernet Switches and the MX Series. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-virtualchassis.txt

  For more information, see *Virtual Chassis MIBs*.

- VLAN MIB—Contains information about prestandard IEEE 802.10 VLANs and their association with LAN emulation clients. The enterprise-specific VLAN MIB is supported only on EX Series Ethernet Switches. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vlan.txt

  For more information, see *VLAN MIB*.

- VPLS MIBs—Provides information about generic, BGP-based, and LDP-based VPLS and pseudowires associated with the VPLS networks. The enterprise-specific VPLS MIBs are Juniper Networks extensions of the following IETF standard MIBs defined in Internet draft draft-ietf-l2vpn-vpls-mib-05.txt, and are implemented as part of the **jnxExperiment** branch:

  - **VPLS-Generic-Draft-01-MIB** implemented as **mib-jnx-vpls-generic.txt**
  - **VPLS-BGP-Draft-01-MIB** implemented as **mib-jnx-vpls-bgp.txt**
  - **VPLS-LDP-Draft-01-MIB** implemented as **mib-jnx-vpls-ldp.txt**

  For downloadable versions of these MIBs, see:

  - http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-generic.txt
  - http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-bgp.txt
  - http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-ldp.txt

- VPN Certificate Objects MIB—Provides support for monitoring the local and CA certificates loaded on the router. This MIB is currently supported by Junos OS for J Series and SRX Series devices only. For a downloadable version of this MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-cert.txt

  For more information, see *VPN Certificate Objects MIB*.

- VPN MIB—Provides monitoring for Layer 3 VPNs, Layer 2 VPNs, and virtual private LAN service (VPLS) (read access only). For a downloadable version of the MIB, see http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpn.txt

  For more information, see *VPN MIB*.

Related Documentation

- Standard SNMP MIBs Supported by Junos OS on page 19
- Juniper Networks Enterprise-Specific MIBs and Supported Devices on page 46
- Loading MIB Files to a Network Management System on page 297

Juniper Networks Enterprise-Specific MIBs and Supported Devices

Table 5 on page 47 lists the enterprise-specific MIBs that are supported on various devices running Junos OS.

NOTE: In this table, a value of 1 in any of the platform columns (ACX, M, MX, T, EX, J, and SRX) denotes that the corresponding MIB is supported on that particular platform. A value of 0 denotes that the MIB is not supported on the platform.

NOTE: This topic uses the following classification for SRX devices: Low-End (SRX100, SRX210, SRX220, and SRX240), Mid-Range (SRX550. SRX650 and SRX1400), and High-End (SRX3400, SRX3600, SRX5600, and SRX5800).

Table 5: Enterprise-Specific MIBs and Supported Devices

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
AAA Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-user-aaa.txt	0	1	1	0	0	0	0	0	1	1
Access Authentication Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-auth.txt	0	0	0	0	0	1	0	1	1	1
Alarm MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis-alarm.txt	1	1	1	1	1	1	1	1	1	1
Analyzer MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-analyzer.txt	0	0	0	0	1	0	0	0	0	0
Antivirus Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-utm-av.txt	0	0	0	0	0	0	0	1	0	0
ATM Class-of-Service MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-atm-cos.txt	0	1	1	1	0	0	0	1	0	1

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
ATM MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-atm.txt	1	1	1	1	0	0	0	0	0	0
BGP4 V2 MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-bgpmib2.txt	1	1	1	1	1	1	1	1	1	1
Bidirectional Forwarding Detection MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-bfd.txt	1	1	1	1	1	1	1	1	1	1
Chassis Forwarding MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis-fwdd.txt	1	0	0	0	0	1	1	1	0	0
Chassis MIBs http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chassis.txt http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-chas-defines.txt	1	1	1	1	1	1	1	1	1	1
Chassis Cluster MIBs http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jsrpd.txt	0	0	0	0	0	0	0	0	1	1
Class-of-Service MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-cos.txt	1	1	1	1	1	1	1	0	0	1
Configuration Management MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-cfgmgmt.txt	1	1	1	1	1	1	1	1	1	1
Destination Class Usage MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dcu.txt	0	1	1	1	0	1	0	0	1	1

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
DHCP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jdhcp.txt	0	1	1	1	0	0	0	0	0	0
DHCPv6 MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-jdhcpv6.txt	0	1	1	1	0	0	0	0	0	0
Digital Optical Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dom.txt	1	1	1	1	1	1	0	1	1	1
DNS Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-dns.txt	0	0	0	0	0	0	0	0	1	1
Dynamic Flow Capture MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-dfc.txt	0	1	1	1	0	0	0	0	0	0
Ethernet MAC MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/jnx-mac.txt	0	1	1	1	1	1	0	0	0	1
Event MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-event.txt	1	1	1	1	1	1	1	1	1	1
EX Series MAC Notification MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ex-mac-notification.txt	0	0	0	0	1	0	0	0	0	0
EX Series SMI MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ex-smi.txt	0	0	0	0	1	0	0	0	0	0

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
Experimental MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-exp.txt	1	1	1	1	1	1	0	0	0	0
Firewall MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-firewall.txt	1	1	1	1	1	1	1	1	1	1
Flow Collection Services MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-coll.txt	0	1	1	1	0	0	0	0	0	0
Host Resources MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-hostresources.txt	1	1	1	1	1	1	0	1	1	1
Interface MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-if-extensions.txt	1	1	1	1	1	1	1	1	1	1
IP Forward MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipforward.txt	1	1	1	1	1	1	1	1	1	1
IPsec Generic Flow Monitoring Object MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipsec-flow-mon.txt	0	0	0	1	0	0	0	1	1	1
IPsec Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipsec-monitor-asp.txt	0	1	1	1	0	1	0	0	0	0
IPsec VPN Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-ipsec-vpn.txt	0	0	0	1	0	0	0	1	1	1

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
IPv4 MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipv4.txt	1	1	1	1	1	1	1	1	1	1
IPv6 and ICMPv6 MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ipv6.txt	0	1	1	1	1	0	1	1	1	1
L2ALD MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2ald.txt	0	0	1	0	1	0	0	0	0	0
L2CP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2cp-features.txt	0	0	0	0	1	0	0	0	0	0
L2TP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-l2tp.txt	0	1	1	0	0	0	0	0	0	0
LDP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ldp.txt	1	1	1	1	0	0	1	0	0	1
License MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-license.txt	0	1	1	1	0	0	0	1	1	1
Logical Systems MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-lsys-securityprofile.txt	0	0	0	0	0	0	0	0	1	1
MIMSTP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mimstp.txt	0	0	1	0	1	0	0	0	0	0

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
MPLS LDP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mpls-ldp.txt	1	1	1	1	1	1	1	0	0	0
MPLS MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-mpls.txt	1	1	1	1	1	1	1	0	0	1
NAT Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-nat.txt	0	0	0	0	0	1	0	1	1	1
NAT Resources-Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/ mibs/mib-jnx-sp-nat.txt	0	1	1	1	0	0	0	0	0	0
OTN Interface Management MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-otn.txt	0	1	1	1	0	0	0	0	0	0
Packet Forwarding Engine MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pfe.txt	1	1	1	1	0	1	1	1	1	1
Packet Mirror MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-packet-mirror.txt	0	0	0	0	1	0	0	0	0	0
PAE Extension MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pae-extension.txt	0	0	0	0	1	0	0	0	0	0
Passive Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pmon.txt	0	1	1	1	0	0	0	0	0	0

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
Ping MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ping.txt	1	1	1	1	1	1	0	1	1	1
Policy Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-policy.txt	0	0	0	0	0	1	0	1	1	1
Power Supply Unit MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-power-supply-unit.txt	0	0	0	0	1	0	1	0	0	0
PPP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-ppp.txt.	0	1	1	0	0	0	0	0	0	0
PPPoE MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pppoe.txt.	0	1	1	0	0	0	0	0	0	0
Pseudowire ATM MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pwatm.txt.	0	1	0	0	1	0	0	0	0	0
Pseudowire TDM MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-pwtdm.txt .	1	1	1	1	0	0	0	0	0	0
Real-Time Performance Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rpm.txt	0	1	1	1	1	1	0	1	0	0
Reverse-Path-Forwarding MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rpf.txt	1	1	1	1	1	1	1	1	1	1

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
RMON Events and Alarms MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rmon.txt	1	1	1	1	1	1	1	1	1	1
RSVP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-rsvp.txt	1	1	1	1	1	0	1	0	0	0
Security Interface Extension Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-if-ext.txt	0	0	0	0	0	1	0	1	1	1
Security Screening Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-screening.txt	0	0	0	0	0	0	0	0	0	1
Services PIC MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sp.txt	0	1	1	1	0	0	0	0	0	0
SNMP IDP MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-idp.txt.	0	0	0	0	0	0	0	1	1	1
SONET APS MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sonetaps.txt	0	1	1	1	0	0	0	0	0	0
SONET/SDH Interface Management MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-sonet.txt	0	1	1	1	0	0	0	0	0	0
Source Class Usage MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-scu.txt	0	1	1	1	0	0	0	0	0	1

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
SPU Monitoring MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-spu-monitoring.txt	0	0	0	0	0	0	0	1	1	1
Structure of Management Information MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-smi.txt	1	1	1	1	1	1	0	1	1	1
Subscriber MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-subscriber.txt	1	0	1	0	0	0	0	0	0	0
System Log MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-syslog.txt	0	1	1	1	1	1	1	1	1	1
Traceroute MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-traceroute.txt	0	1	1	1	1	1	0	1	1	1
Utility MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-util.txt	0	1	1	1	1	1	0	1	1	1
Virtual Chassis MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-virtualchassis.txt	0	0	0	0	1	1	0	0	0	0
VLAN MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vlan.txt	0	0	0	0	1	0	0	0	0	0

Table 5: Enterprise-Specific MIBs and Supported Devices *(continued)*

Enterprise-Specific MIB	Platforms									
								SRX		
	ACX	M	T	J	MX	EX	PTX	Low-End	Mid-Range	High-End
VPLS MIBs • http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-generic.txt • http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-ldp.txt • http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpls-bgp.txt	0	1	1	1	1	0	0	0	0	0
VPN Certificate Objects MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-js-cert.txt	0	0	0	0	0	1	0	1	1	1
VPN MIB http://www.juniper.net/techpubs/en_US/junos13.2/topics/reference/mibs/mib-jnx-vpn.txt	1	1	1	1	1	1	0	0	0	0

Related Documentation

- Juniper Networks Enterprise-Specific MIBs on page 35
- Juniper Networks Enterprise-Specific SNMP Traps on page 63
- Standard SNMP MIBs Supported by Junos OS on page 19
- Loading MIB Files to a Network Management System on page 297

SNMP MIB Objects Supported by Junos OS for the Set Operation

The following table lists the SNMP MIB objects that are supported for the **snmp set** operation by Junos OS.

Object Name	Object Identifier
RFC 1907	
sysContact	**1.3.6.1.2.1.1.4**
sysName	**1.3.6.1.2.1.1.5**
sysLocation	**1.3.6.1.2.1.1.6**
snmpEnableAuthenTraps	**1.3.6.1.2.1.11.30**

Object Name	Object Identifier
RFC 2819a	
alarmInterval	1.3.6.1.2.1.16.3.1.1.2
alarmVariable	1.3.6.1.2.1.16.3.1.1.2
alarmSampleType	1.3.6.1.2.1.16.3.1.1.4
alarmStartupAlarm	1.3.6.1.2.1.16.3.1.1.6
alarmRisingThreshold	1.3.6.1.2.1.16.3.1.1.7
alarmFallingThreshold	1.3.6.1.2.1.16.3.1.1.8
alarmRisingEventIndex	1.3.6.1.2.1.16.3.1.1.9
alarmFallingEventIndex	1.3.6.1.2.1.16.3.1.1.10
alarmOwner	1.3.6.1.2.1.16.3.1.1.11
alarmStatus	1.3.6.1.2.1.16.3.1.1.12
eventDescription	1.3.6.1.2.1.16.9.1.1.2
eventType	1.3.6.1.2.1.16.9.1.1.3
eventCommunity	1.3.6.1.2.1.16.9.1.1.4
eventOwner	1.3.6.1.2.1.16.9.1.1.6
eventStatus	1.3.6.1.2.1.16.9.1.1.7
RFC 2925a	
pingMaxConcurrentRequests	1.3.6.1.2.1.80.1.1
pingCtlTargetAddressType	1.3.6.1.2.1.80.1.2.1.3
pingCtlTargetAddress	1.3.6.1.2.1.80.1.2.1.4
pingCtlDataSize	1.3.6.1.2.1.80.1.2.1.5
pingCtlTimeOut	1.3.6.1.2.1.80.1.2.1.6
pingCtlProbeCount	1.3.6.1.2.1.80.1.2.1.7
pingCtlAdminStatus	1.3.6.1.2.1.80.1.2.1.8

Object Name	Object Identifier
pingCtlDataFill	1.3.6.1.2.1.80.1.2.1.9
pingCtlFrequency	1.3.6.1.2.1.80.1.2.1.10
pingCtlMaxRows	1.3.6.1.2.1.80.1.2.1.11
pingCtlStorageType	1.3.6.1.2.1.80.1.2.1.12
pingCtlTrapGeneration	1.3.6.1.2.1.80.1.2.1.13
pingCtlTrapProbeFailureFilter	1.3.6.1.2.1.80.1.2.1.14
pingCtlTrapTestFailureFilter	1.3.6.1.2.1.80.1.2.1.15
pingCtlType	1.3.6.1.2.1.80.1.2.1.16
pingCtlDescr	1.3.6.1.2.1.80.1.2.1.17
pingCtlSourceAddressType	1.3.6.1.2.1.80.1.2.1.18
pingCtlSourceAddress	1.3.6.1.2.1.80.1.2.1.19
pingCtlIfIndex	1.3.6.1.2.1.80.1.2.1.20
pingCtlByPassRouteTable	1.3.6.1.2.1.80.1.2.1.21
pingCtlDSField	1.3.6.1.2.1.80.1.2.1.22
pingCtlRowStatus	1.3.6.1.2.1.80.1.2.1.23
RFC 2925B	
traceRouteMaxConcurrentRequests	1.3.6,1.2.1.81.1.1
traceRouteCtlTargetAddressType	1.3.6,1.2.1.81.1.2.1.3
traceRouteCtlTargetAddress	1.3.6,1.2.1.81.1.2.1.4
traceRouteCtlByPassRouteTable	1.3.6,1.2.1.81.1.2.1.5
traceRouteCtlDataSize	1.3.6,1.2.1.81.1.2.1.6
traceRouteCtlTimeOut	1.3.6,1.2.1.81.1.2.1.7
traceRouteCtlProbesPerHop	1.3.6,1.2.1.81.1.2.1.8
traceRouteCtlPort	1.3.6,1.2.1.81.1.2.1.9

Object Name	Object Identifier
traceRouteCtlMaxTtl	1.3.6,1.2.1.81.1.2.1.10
traceRouteCtlDSField	1.3.6,1.2.1.81.1.2.1.11
traceRouteCtlSourceAddressType	1.3.6,1.2.1.81.1.2.1.12
traceRouteCtlSourceAddress	1.3.6,1.2.1.81.1.2.1.13
traceRouteCtlIfIndex	1.3.6,1.2.1.81.1.2.1.14
traceRouteCtlMiscOptions	1.3.6,1.2.1.81.1.2.1.15
traceRouteCtlMaxFailure	1.3.6,1.2.1.81.1.2.1.16
traceRouteCtlDontFragment	1.3.6,1.2.1.81.1.2.1.17
traceRouteCtlInitialTtl	1.3.6,1.2.1.81.1.2.1.18
traceRouteCtlFrequency	1.3.6,1.2.1.81.1.2.1.19
traceRouteCtlStorageType	1.3.6,1.2.1.81.1.2.1.20
traceRouteCtlAdminStatus	1.3.6,1.2.1.81.1.2.1.21
traceRouteCtlDescr	1.3.6,1.2.1.81.1.2.1.22
traceRouteCtlMaxRows	1.3.6,1.2.1.81.1.2.1.23
traceRouteCtlTrapGeneration	1.3.6,1.2.1.81.1.2.1.24
traceRouteCtlCreateHopEntries	1.3.6,1.2.1.81.1.2.1.25
traceRouteCtlType	1.3.6,1.2.1.81.1.2.1.26
traceRouteCtlRowStatus	1.3.6,1.2.1.81.1.2.1.27
Enterprise-Specific PING MIB	
jnxPingCtlIfName	1.3.6.1.4.1.2636.3.7.1.2.1.3
jnxPingCtlRoutingIfIndex	1.3.6.1.4.1.2636.3.7.1.2.1.4
jnxPingCtlRoutingIfName	1.3.6.1.4.1.2636.3.7.1.2.1.5
jnxPingCtlRoutingInstanceName	1.3.6.1.4.1.2636.3.7.1.2.1.6
jnxPingCtlRttThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.7

Object Name	Object Identifier
jnxPingCtlRttStdDevThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.8
jnxPingCtlRttJitterThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.9
jnxPingCtlEgressTimeThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.10
jnxPingCtlEgressStdDevThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.11
jnxPingCtlEgressJitterThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.12
jnxPingCtlIngressTimeThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.13
jnxPingCtlIngressStdDevThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.14
jnxPingCtlIngressJitterThreshold	1.3.6.1.4.1.2636.3.7.1.2.1.15
jnxPingTrapGeneration	1.3.6.1.4.1.2636.3.7.1.2.1.16
Enterprise-Specific Traceroute MIB	
jnxTRCtlIfName	1.3.6.1.4.1.2636.3.8.1.2.1.3
jnxTRCtlRoutingInstanceName	1.3.6.1.4.1.2636.3.8.1.2.1.4
RFC 3413 Target MIB	
snmpTargetSpinLock	1.3.6.1.6.3.12.1.1
snmpTargetAddrTDomain	1.3.6.1.6.3.12.1.2.1.2
snmpTargetAddrTAddress	1.3.6.1.6.3.12.1.2.1.3
snmpTargetAddrTimeout	1.3.6.1.6.3.12.1.2.1.4
snmpTargetAddrRetryCount	1.3.6.1.6.3.12.1.2.1.5
snmpTargetAddrTagList	1.3.6.1.6.3.12.1.2.1.6
snmpTargetAddrParams	1.3.6.1.6.3.12.1.2.1.7
snmpTargetAddrStorageType	1.3.6.1.6.3.12.1.2.1.8
snmpTargetAddrRowStatus	1.3.6.1.6.3.12.1.2.1.9
snmpTargetParamsMPModel	1.3.6.1.6.3.12.1.3.1.2
snmpTargetParamsSecurityModel	1.3.6.1.6.3.12.1.3.1.3

Object Name	Object Identifier
snmpTargetParamsSecurityLevel	1.3.6.1.6.3.12.1.3.1.4
snmpTargetParamsSecurityName	1.3.6.1.6.3.12.1.3.1.5
snmpTargetParamsStorageType	1.3.6.1.6.3.12.1.3.1.6
snmpTargetParamsRowStatus	1.3.6.1.6.3.12.1.3.1.7
RFC 3413 Notify MIB	
snmpNotifyTag	1.3.6.1.6.3.13.1.1.1.2
snmpNotifyType	1.3.6.1.6.3.13.1.1.1.3
snmpNotifyStorageType	1.3.6.1.6.3.13.1.1.1.4
snmpNotifyRowStatus	1.3.6.1.6.3.13.1.1.1.5
snmpNotifyFilterProfileName	1.3.6.1.6.3.13.1.2.1.1
snmpNotifyFilterProfileStorType	1.3.6.1.6.3.13.1.2.1.2
snmpNotifyFilterProfileRowStatus	1.3.6.1.6.3.13.1.2.1.3
snmpNotifyFilterMask	1.3.6.1.6.3.13.1.3.1.2
snmpNotifyFilterType	1.3.6.1.6.3.13.1.3.1.3
snmpNotifyFilterStorageType	1.3.6.1.6.3.13.1.3.1.4
snmpNotifyFilterRowStatus	1.3.6.1.6.3.13.1.3.1.5
RFC 2574	
usmUserSpinLock	1.3.6.1.6.3.15.1.2.1
usmUserCloneFrom	1.3.6.1.6.3.15.1.2.2.1.4
usmUserAuthProtocol	1.3.6.1.6.3.15.1.2.2.1.5
usmUserAuthKeyChange	1.3.6.1.6.3.15.1.2.2.1.6
usmUserOwnAuthKeyChange	1.3.6.1.6.3.15.1.2.2.1.7
usmUserPrivProtocol	1.3.6.1.6.3.15.1.2.2.1.8
usmUserPrivKeyChange	1.3.6.1.6.3.15.1.2.2.1.9

Object Name	Object Identifier
usmUserOwnPrivKeyChange	1.3.6.1.6.3.15.1.2.2.1.10
usmUserPublic	1.3.6.1.6.3.15.1.2.2.1.11
usmUserStorageType	1.3.6.1.6.3.15.1.2.2.1.12
usmUserStatus	1.3.6.1.6.3.15.1.2.2.1.13
RFC 2575	
vacmGroupName	1.3.6.1.6.3.16.1.2.1.3
vacmSecurityToGroupStorageType	1.3.6.1.6.3.16.1.2.1.4
vacmSecurityToGroupStatus	1.3.6.1.6.3.16.1.2.1.5
vacmAccessContextMatch	1.3.6.1.6.3.16.1.4.1.4
vacmAccessReadViewName	1.3.6.1.6.3.16.1.4.1.5
vacmAccessWriteViewName	1.3.6.1.6.3.16.1.4.1.6
vacmAccessNotifyViewName	1.3.6.1.6.3.16.1.4.1.7
vacmAccessStorageType	1.3.6.1.6.3.16.1.4.1.8
vacmAccessStatus	1.3.6.1.6.3.16.1.4.1.9
vacmViewSpinLock	1.3.6.1.6.3.16.1.5.1
vacmViewTreeFamilyMask	1.3.6.1.6.3.16.1.5.2.1.3
vacmViewTreeFamilyType	1.3.6.1.6.3.16.1.5.2.1.4
vacmViewTreeFamilyStorageType	1.3.6.1.6.3.16.1.5.2.1.5
vacmViewTreeFamilyStatus	1.3.6.1.6.3.16.1.5.2.1.6
RFC 2576	
snmpCommunityName	1.3.6.1.6.3.18.1.1.1.2
snmpCommunitySecurityName	1.3.6.1.6.3.18.1.1.1.3
snmpCommunityContextEngineID	1.3.6.1.6.3.18.1.1.1.4
snmpCommunityContextName	1.3.6.1.6.3.18.1.1.1.5

Object Name	Object Identifier
snmpCommunityTransportTag	1.3.6.1.6.3.18.1.1.1.6
snmpCommunityStorageType	1.3.6.1.6.3.18.1.1.1.7
snmpCommunityStatus	1.3.6.1.6.3.18.1.1.1.8
RFC 2576	
snmpTargetAddrMask	1.3.6.1.6.3.18.1.2.1.1
snmpTargetAddrMMS	1.3.6.1.6.3.18.1.2.1.2

Related Documentation

- Standard SNMP MIBs Supported by Junos OS on page 19
- Juniper Networks Enterprise-Specific MIBs on page 35
- Juniper Networks Enterprise-Specific MIBs and Supported Devices on page 46

Standard SNMP Traps Supported on Devices Running Junos OS

This topic provides pointers to the standard SNMP traps supported by Junos OS.

NOTE: For scalability reasons, the MPLS traps are generated by the ingress router only. For information about disabling the generation of MPLS traps, see the *Junos OS MPLS Applications Library for Routing Devices* document.

- *Standard SNMP Version 1 Traps*
- *Standard SNMP Version 2 Traps*
- *Standard SNMP Traps on EX Series Ethernet Switches*
- *Unsupported Standard SNMP Traps*

Related Documentation

- Juniper Networks Enterprise-Specific SNMP Traps on page 63
- Juniper Networks Enterprise-Specific MIBs on page 35
- Standard SNMP MIBs Supported by Junos OS on page 19
- Configuring SNMP Trap Options and Groups on a Device Running Junos OS on page 100
- *Managing Traps and Informs*

Juniper Networks Enterprise-Specific SNMP Traps

This topic provides pointers to the enterprise-specific SNMP traps supported by the Junos OS.

NOTE: All enterprise-specific SNMP traps supported by the Junos OS can be sent in version 1, 2, and 3 formats.

- *Juniper Networks Enterprise-Specific SNMP Version 1 Traps*
- *Juniper Networks Enterprise-Specific SNMP Version 2 Traps*
- *Juniper Networks Enterprise-Specific BGP Traps*
- *Juniper Networks Enterprise-Specific LDP Traps*
- *Juniper Networks Enterprise-Specific License MIB Notifications*
- *Juniper Networks Enterprise-Specific MIMSTP Traps*
- *Juniper Networks Enterprise-Specific MPLS Traps*

NOTE: For scalability reasons, the MPLS traps are generated by the ingress router only. For information about disabling the generation of MPLS traps, see the *Junos OS MPLS Applications Library for Routing Devices*.

- *Juniper Networks Enterprise-Specific Traps on EX Series Switches*
- *Juniper Networks Enterprise-Specific Traps on MX Series 3D Universal Edge Routers*

Related Documentation

- Standard SNMP Traps Supported on Devices Running Junos OS on page 63
- Juniper Networks Enterprise-Specific MIBs on page 35
- Standard SNMP MIBs Supported by Junos OS on page 19
- Configuring SNMP Trap Options and Groups on a Device Running Junos OS on page 100
- *Managing Traps and Informs*

CHAPTER 4

Remote Monitoring (RMON), Health Monitoring, and Service Quality Monitoring

Understanding RMON Alarms

An RMON alarm identifies:

- A specific MIB object that is monitored.
- The frequency of sampling.
- The method of sampling.
- The thresholds against which the monitored values are compared.

An RMON alarm can also identify a specific **eventTable** entry to be triggered when a threshold is crossed.

Configuration and operational values are defined in **alarmTable** in RFC 2819. Additional operational values are defined in Juniper Networks enterprise-specific extensions to **alarmTable** (**jnxRmonAlarmTable**).

This topic covers the following sections:

alarmTable

alarmTable in the RMON MIB allows you to monitor and poll the following:

- **alarmIndex**—The index value for **alarmTable** that identifies a specific entry.
- **alarmInterval**—The interval, in seconds, over which data is sampled and compared with the rising and falling thresholds.
- **alarmVariable**—The MIB variable that is monitored by the alarm entry.
- **alarmSampleType**—The method of sampling the selected variable and calculating the value to be compared against the thresholds.
- **alarmValue**—The value of the variable during the last sampling period. This value is compared with the rising and falling thresholds.
- **alarmStartupAlarm**—The alarm sent when the entry is first activated.
- **alarmRisingThreshold**—The upper threshold for the sampled variable.
- **alarmFallingThreshold**—The lower threshold for the sampled variable.
- **alarmRisingEventIndex**—The **eventTable** entry used when a rising threshold is crossed.
- **alarmFallingEventIndex**—The **eventTable** entry used when a falling threshold is crossed.
- **alarmStatus**—Method for adding and removing entries from the table. It can also be used to change the state of an entry to allow modifications.

NOTE: **If this object is not set to valid, the associated event alarm does not take any action.**

jnxRmonAlarmTable

The **jnxRmonAlarmTable** is a Juniper Networks enterprise-specific extension to **alarmTable**. It provides additional operational information and includes the following objects:

- **jnxRmonAlarmGetFailCnt**—The number of times the internal **Get** request for the variable monitored by this entry has failed.
- **jnxRmonAlarmGetFailTime**—The value of **sysUpTime** when an internal **Get** request for the variable monitored by this entry last failed.
- **jnxRmonAlarmGetFailReason**—The reason an internal **Get** request for the variable monitored by this entry last failed.
- **jnxRmonAlarmGetOkTime**—The value of **sysUpTime** when an internal **Get** request for the variable monitored by this entry succeeded and the entry left the **getFailure** state.
- **jnxRmonAlarmState**—The current state of this RMON alarm entry.

To view the Juniper Networks enterprise-specific extensions to the RMON Events and Alarms and Event MIB, see http://www.juniper.net/techpubs/en_US/junos10.3/topics/reference/mibs/mib-jnx-rmon.txt.

For more information about the Juniper Networks enterprise-specific extensions to the RMON Events and Alarms MIB, see "*RMON Events and Alarms MIB*" in the *SNMP MIBs and Traps Reference*.

Related Documentation

- Understanding RMON Events on page 76
- Configuring an Alarm Entry and Its Attributes on page 158
- Using alarmTable to Monitor MIB Objects on page 318

MIB Support Details

Table 6 on page 67 shows enterprise-specific MIB objects supported by Junos OS and provides notes detailing how they are handled when a routing instance is specified in an SNMP request. An en dash (–) indicates that the item is not applicable.

Table 6: MIB Support for Routing Instances (Juniper Networks MIBs)

Object	Support Class	Description/Notes
jnxProducts(1)	–	Product Object IDs
jnxServices(2)	–	Services
jnxMibs(3) **jnxBoxAnatomy(1)**	Class 3	Objects are exposed only for the default logical system.
mpls(2)	Class 2	All instances within a logical system are exposed. Data will not be segregated down to the routing instance level.
ifJnx(3)	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxAlarms(4)	Class 3	Objects are exposed only for the default logical system.
jnxFirewalls(5)	Class 4	Data is not segregated by routing instance. All instances are exposed.
jnxDCUs(6)	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxPingMIB(7)	Class 3	Objects are exposed only for the default logical system.
jnxTraceRouteMIB(8)	Class 3	Objects are exposed only for the default logical system.

Table 6: MIB Support for Routing Instances (Juniper Networks MIBs) *(continued)*

Object	Support Class	Description/Notes
jnxATM(10)	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxIpv6(11)	Class 4	Data is not segregated by routing instance. All instances are exposed.
jnxIpv4(12)	Class 1	**jnxIpv4AddrTable(1)**. Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxRmon(13)	Class 3	**jnxRmonAlarmTable(1)**. Objects are exposed only for the default logical system.
jnxLdp(14)	Class 2	**jnxLdpTrapVars(1)**. All instances within a logical system are exposed. Data will not be segregated down to the routing instance level.
jnxCos(15) **jnxCosIfqStatsTable(1)** **jnxCosFcTable(2)** **jnxCosFcIdTable(3)** **jnxCosQstatTable(4)**	Class 3	Objects are exposed only for the default logical system.
jnxScu(16) **jnxScuStatsTable(1)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxRpf(17) **jnxRpfStatsTable(1)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxCfgMgmt(18)	Class 3	Objects are exposed only for the default logical system.
jnxPMon(19) **jnxPMonFlowTable(1)** **jnxPMonErrorTable(2)** **jnxPMonMemoryTable(3)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxSonet(20) **jnxSonetAlarmTable(1)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.

Table 6: MIB Support for Routing Instances (Juniper Networks MIBs) *(continued)*

Object	Support Class	Description/Notes
jnxAtmCos(21) **jnxCosAtmVcTable(1)** **jnxCosAtmScTable(2)** **jnxCosAtmVcQstatsTable(3)** **jnxCosAtmTrunkTable(4)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
ipSecFlowMonitorMIB(22)	–	–
jnxMac(23) **jnxMacStats(1)**	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
apsMIB(24)	Class 3	Objects are exposed only for the default logical system.
jnxChassisDefines(25)	Class 3	Objects are exposed only for the default logical system.
jnxVpnMIB(26)	Class 2	All instances within a logical system are exposed. Data will not be segregated down to the routing instance level.
jnxSericesInfoMib(27)	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxCollectorMIB(28)	Class 1	Only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.
jnxHistory(29)	–	–
jnxSpMIB(32)	Class 3	Objects are exposed only for the default logical system.

Table 7 on page 70 shows Class 1 MIB objects (standard and enterprise-specific MIBs) supported by Junos OS. With Class 1 objects, only those logical interfaces (and their parent physical interfaces) that belong to a specific routing instance are exposed.

Table 7: Class 1 MIB Objects (Standard and Juniper MIBs)

Class	MIB	Objects
Class 1	**802.3ad.mib**	(**dot3adAgg**) MIB objects: **dot3adAggTable** **dot3adAggPortListTable** (**dot3adAggPort**) **dot3adAggPortTable** **dot3adAggPortStatsTable** **dot3adAggPortDebugTable**
	rfc2863a.mib	**ifTable** **ifXTable** **ifStackTable**
	rfc2011a.mib	**ipAddrTable** **ipNetToMediaTable**
	rtmib.mib	**ipForward** (**ipCidrRouteTable**)
	rfc2665a.mib	**dot3StatsTable** **dot3ControlTable** **dot3PauseTable**
	rfc2495a.mib	**dsx1ConfigTable** **dsx1CurrentTable** **dsx1IntervalTable** **dsx1TotalTable** **dsx1FarEndCurrentTable** **dsx1FarEndIntervalTable** **dsx1FarEndTotalTable** **dsx1FracTable** ...
	rfc2496a.mib	**dsx3 (dsx3ConfigTable)**
	rfc2115a.mib	**frDlcmiTable** (and related MIB objects)
	rfc3592.mib	**sonetMediumTable** (and related MIB objects)

Table 7: Class 1 MIB Objects (Standard and Juniper MIBs) *(continued)*

Class	MIB	Objects
	rfc3020.mib	**mfrMIB** **mfrBundleTable** **mfrMibBundleLinkObjects** **mfrBundleIfIndexMappingTable** (and related MIB objects)
	ospf2mib.mib	All objects
	ospf2trap.mib	All objects
	bgpmib.mib	All objects
	rfc2819a.mib	Example: **etherStatsTable**

Table 7: Class 1 MIB Objects (Standard and Juniper MIBs) *(continued)*

Class	MIB	Objects
Class 1	**rfc2863a.mib**	Examples: **ifXtable** **ifStackTable**
	rfc2665a.mib	**etherMIB**
	rfc2515a.mib	**atmMIB** objects Examples: **atmInterfaceConfTable** **atmVplTable** **atmVclTable**
	rfc2465.mib	**ip-v6mib** Examples: **ipv6IfTable** **ipv6AddrPrefixTable** **ipv6NetToMediaTable** **ipv6RouteTable**
	rfc2787a.mib	**vrrp mib**
	rfc2932.mib	**ipMRouteMIB** **ipMRouteStdMIB**
	mroutemib.mib	**ipMRoute1MIBObjects**
	isismib.mib	**isisMIB**
	pimmib.mib	**pimMIB**
	msdpmib.mib	**msdpmib**
	jnx-if-extensions.mib	Examples: **ifJnxTable** **ifChassisTable**
	jnx-dcu.mib	**jnxDCUs**
	jnx-atm.mib	

Table 7: Class 1 MIB Objects (Standard and Juniper MIBs) *(continued)*

Class	MIB	Objects
		Examples: **jnxAtmIfTable** **jnxAtmVCTable** **jnxAtmVpTable**
	jnx-ipv4.mib	**jnxipv4** Example: **jnxIpv4AddrTable**
	jnx-cos.mib	Examples: **jnxCosIfqStatsTable** **jnxCosQstatTable**
	jnx-scu.mib	Example: **jnxScuStatsTable**
	jnx-rpf.mib	Example: **jnxRpfStatsTable**
	jnx-pmon.mib	Example: **jnxPMonFlowTable**
	jnx-sonet.mib	Example: **jnxSonetAlarmTable**
Class 1	**jnx-atm-cos.mib**	Examples: **jnxCosAtmVcTable** **jnxCosAtmVcScTable** **jnxCosAtmVcQstatsTable** **jnxCosAtmTrunkTable**
	jnx-mac.mib	Example: **jnxMacStatsTable**
	jnx-services.mib	Example: **jnxSvcFlowTableAggStatsTable**
	jnx-coll.mib	**jnxCollectorMIB** Examples: **jnxCollPicIfTable** **jnxCollFileEntry**

Table 8 on page 74 shows Class 2 MIB objects (standard and enterprise-specific MIBs) supported by Junos OS. With Class 2 objects, all instances within a logical system are exposed. Data will not be segregated down to the routing instance level.

Table 8: Class 2 MIB Objects (Standard and Juniper MIBs)

Class	MIB	Objects
Class 2	**rfc3813.mib**	**mplsLsrStdMIB** Examples: **mplsInterfaceTable** **mplsInSegmentTable** **mplsOutSegmentTable** **mplsLabelStackTable** **mplsXCTable** (and related MIB objects)
	igmpmib.mib	**igmpStdMIB** NOTE: The **igmpmib.mib** is the draft version of the IGMP Standard MIB in the experimental tree. Junos OS does not support the original IGMP Standard MIB.
	l3vpnmib.mib	**mplsVpnmib**
	jnx-mpls.mib	Example: **mplsLspList**
	jnx-ldp.mib	**jnxLdp** Example: **jnxLdpStatsTable**
	jnx-vpn.mib	**jnxVpnMIB**
	jnx-bgpmib2.mib	**jnxBgpM2Experiment**

Table 9 on page 75 shows Class 3 MIB objects (standard and enterprise-specific MIBs) supported by Junos OS. With Class 3, objects are exposed only for the default logical system.

Table 9: Class 3 MIB Objects (Standard and Juniper MIBs)

Class	MIB	Objects
Class 3	rfc2819a.mib	rmonEvents alarmTable logTable eventTable agentxMIB
	rfc2925a.mib	pingmib
	rfc2925b.mib	tracerouteMIB
	jnxchassis.mib	jnxBoxAnatomy
	jnx-chassis-alarm.mib	jnxAlarms
	jnx-ping.mib	jnxPingMIB
	jnx-traceroute.mib	jnxTraceRouteMIB
	jnx-rmon.mib	jnxRmonAlarmTable
	jnx-cos.mib	Example: jnxCosFcTable
	jnx-cfgmgmt.mib	Example: jnxCfgMgmt
	jnx-sonetaps.mib	apsMIBObjects
	jnx-sp.mib	jnxSpMIB
	ggsn.mib	ejnmobileipABmib
	rfc1907.mib	snmpModules
	snmpModules	Examples: snmpMIB snmpFrameworkMIB

Table 10 on page 76 shows Class 4 MIB objects (standard and enterprise-specific MIBs) supported by Junos OS. With Class 4 objects, data is not segregated by routing instance. All instances are exposed.

Table 10: Class 4 MIB Objects (Standard and Juniper MIBs)

Class	MIB	Objects
Class 4	**system**	Example: **sysORTable**
	rfc2011a.mib	**ip (ipDefaultTTL, ipInReceives)** **icmp**
	rfc2012a.mib	**tcp** **tcpConnTable** **ipv6TcpConnTable**
	rfc2013a.mib	**udp** **udpTable** **ipv6UdpTable**
	rfc2790a.mib	**hrSystem**
	rfc2287a.mib	**sysApplOBJ**
	jnx-firewall.mib	**jnxFirewalls**
	jnx-ipv6.mib	**jnxIpv6**

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Support Classes for MIB Objects on page 17
- Trap Support for Routing Instances on page 18

Understanding RMON Events

An RMON event allows you to log the crossing of thresholds of other MIB objects. It is defined in **eventTable** for the RMON MIB.

This section covers the following topics:

- eventTable on page 76

eventTable

eventTable contains the following objects:

- **eventIndex**—An index that uniquely identifies an entry in **eventTable**. Each entry defines one event that is generated when the appropriate conditions occur.
- **eventDescription**—A comment describing the event entry.

- **eventType**—Type of notification that the probe makes about this event.
- **eventCommunity**—Trap group used if an SNMP trap is to be sent. If **eventCommunity** is not configured, a trap is sent to each trap group configured with the **rmon-alarm** category.
- **eventLastTimeSent**—Value of **sysUpTime** when this event entry last generated an event.
- **eventOwner**—Any text string specified by the creating management application or the command-line interface (CLI). Typically, it is used to identify a network manager (or application) and can be used for fine access control between participating management applications.
- **eventStatus**—Status of this event entry.

NOTE: **If this object is not set to valid, no action is taken by the associated event entry. When this object is set to valid, all previous log entries associated with this entry (if any) are deleted.**

Related Documentation

- Understanding RMON Alarms on page 65
- Configuring an Event Entry and Its Attributes on page 162
- Using eventTable to Log Alarms on page 322

Using the Ping MIB for Remote Monitoring Devices Running Junos OS

A ping test is used to determine whether packets sent from the local host reach the designated host and are returned. If the designated host can be reached, the ping test provides the approximate round-trip time for the packets. Ping test results are stored in **pingResultsTable** and **pingProbeHistoryTable**.

RFC 2925 is the authoritative description of the Ping MIB in detail and provides the ASN.1 MIB definition of the Piing MIB.

Related Documentation

- SNMP Remote Operations Overview on page 13
- Starting a Ping Test on page 303
- Monitoring a Running Ping Test on page 304
- Gathering Ping Test Results on page 307
- Stopping a Ping Test on page 309
- Interpreting Ping Variables on page 309

Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS

A traceroute test approximates the path packets take from the local host to the remote host.

RFC 2925 is the authoritative description of the Traceroute MIB in detail and provides the ASN.1 MIB definition of the Traceroute MIB.

Related Documentation

- SNMP Remote Operations Overview on page 13
- Starting a Traceroute Test on page 310
- Monitoring a Running Traceroute Test on page 311
- Monitoring Traceroute Test Completion on page 315
- Gathering Traceroute Test Results on page 316
- Stopping a Traceroute Test on page 317
- Interpreting Traceroute Variables on page 318

Understanding Measurement Points, Key Performance Indicators, and Baseline Values

This chapter topic provides guidelines for monitoring the service quality of an IP network. It describes how service providers and network administrators can use information provided by Juniper Networks routers to monitor network performance and capacity. You should have a thorough understanding of the SNMP and the associated MIB supported by Junos OS.

NOTE: For a good introduction to the process of monitoring an IP network, see RFC 2330, *Framework for IP Performance Metrics*.

This topic contains the following sections:

- Measurement Points on page 78
- Basic Key Performance Indicators on page 79
- Setting Baselines on page 79

Measurement Points

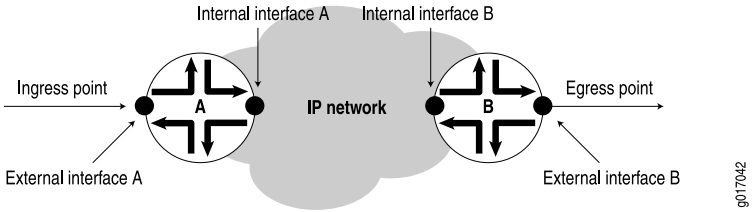
Defining the measurement points where metrics are measured is equally as important as defining the metrics themselves. This section describes measurement points within the context of this chapter and helps identify where measurements can be taken from a service provider network. It is important to understand exactly where a measurement point is. Measurement points are vital to understanding the implication of what the actual measurement means.

An IP network consists of a collection of routers connected by physical links that are all running the Internet Protocol. You can view the network as a collection of routers with an ingress (entry) point and an egress (exit) point. See Figure 2 on page 79.

- Network-centric measurements are taken at measurement points that most closely map to the ingress and egress points for the network itself. For example, to measure delay across the provider network from Site A to Site B, the measurement points should be the ingress point to the provider network at Site A and the egress point at Site B.

- Router-centric measurements are taken directly from the routers themselves, but be careful to ensure that the correct router subcomponents have been identified in advance.

Figure 2: Network Entry Points



NOTE: Figure 2 on page 79 does not show the client networks at customer premises, but they would be located on either side of the ingress and egress points. Although this chapter does not discuss how to measure network services as perceived by these client networks, you can use measurements taken for the service provider network as input into such calculations.

Basic Key Performance Indicators

For example, you could monitor a service provider network for three basic key performance indicators (KPIs):

- *Availability* measures the "reachability" of one measurement point from another measurement point at the network layer (for example, using ICMP ping). The underlying routing and transport infrastructure of the provider network will support the availability measurements, with failures highlighted as unavailability.
- *Health* measures the number and type of errors that are occurring on the provider network, and can consist of both router-centric and network-centric measurements, such as hardware failures or packet loss.
- *Performance* of the provider network measures how well it can support IP services (for example, in terms of delay or utilization).

Setting Baselines

How well is the provider network performing? We recommend an initial three-month period of monitoring to identify a network's normal operational parameters. With this information, you can recognize exceptions and identify abnormal behavior. You should continue baseline monitoring for the lifetime of each measured metric. Over time, you must be able to recognize performance trends and growth patterns.

Within the context of this chapter, many of the metrics identified do not have an allowable operational range associated with them. In most cases, you cannot identify the allowable operational range until you have determined a baseline for the actual variable on a specific network.

Related Documentation

- Understanding RMON for Monitoring Service Quality on page 324
- Defining and Measuring Network Availability on page 328
- Measuring Health on page 334
- Measuring Performance on page 340

CHAPTER 5

Accounting Options and Source Class Usage and Destination Class Usage Options

Accounting Options Overview

An accounting profile represents common characteristics of collected accounting data, including the following:

- Collection interval
- File to contain accounting data
- Specific fields and counter names on which to collect statistics

You can configure multiple accounting profiles, as described in Table 11 on page 81.

Table 11: Types of Accounting Profiles

Type of Profile	Description
Interface profile	Collects the specified error and statistic information.
Filter profile	Collects the byte and packet counts for the counter names specified in the filter profile.
MIB profile	Collects selected MIB statistics and logs them to a specified file.
Routing Engine profile	Collects selected Routing Engine statistics and logs them to a specified file.
Class usage profile	Collects class usage statistics and logs them to a specified file.

Related Documentation

- Understanding Device Management Functions in Junos OS on page 3
- Accounting Options Configuration on page 168
- Configuring Accounting-Data Log Files on page 172
- Configuring the Interface Profile on page 175
- Configuring the Filter Profile on page 177
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167

Understanding Source Class Usage and Destination Class Usage Options

You can maintain packet counts based on the entry and exit points for traffic passing through your network. Entry and exit points are identified by source and destination prefixes grouped into disjoint sets defined as source classes and *destination classes*. You can define classes based on a variety of parameters, such as routing neighbors, autonomous systems, and route filters.

Source class usage (SCU) counts packets sent to customers by performing lookups on the IP source address and the IP destination address. SCU makes it possible to track traffic originating from specific prefixes on the provider core and destined for specific prefixes on the customer edge. You must enable SCU accounting on both the inbound and outbound physical interfaces.

Destination class usage (DCU) counts packets from customers by performing lookups of the IP destination address. DCU makes it possible to track traffic originating from the customer edge and destined for specific prefixes on the provider core router.

On T Series Core Routers and M320 Multiservice Edge Routers, the source class and destination classes are not carried across the platform fabric. The implications of this are as follows:

- On T Series and M320 routers, SCU and DCU accounting is performed before the packet enters the fabric.
- On T Series and M320 routers, DCU is performed before output filters are evaluated. On M Series platforms, DCU is performed after output filters are evaluated.
- If an output filter drops traffic on M Series devices, the dropped packets are excluded from DCU statistics. If an output filter drops traffic on T Series and M320 routers, the dropped packets are included in DCU statistics.

On Enhanced Scaling FPCs (T640-FPC1-ES, T640-FPC2-ES, T640-FPC3-ES, T640-FPC4-1P-ES , and T1600-FPC4-ES), the source class accounting is performed at ingress. On a T4000 Type 5 FPC, the source class accounting is performed at egress. The implications of this are as follows:

- SCU accounting is *not* performed when packets traverse from T4000 Type 5 FPC (ingress FPC) to Enhanced Scaling FPCs (egress FPC).
- SCU accounting is performed when packets traverse from Enhanced Scaling FPCs (ingress FPC) to T4000 Type 5 FPC (egress FPC).

Class-based filter match conditions are not supported on J Series Services Routers.

For more information about source class usage, see the *Routing Policy Feature Guide for Routing Devices*, the *Junos OS Network Interfaces Library for Routing Devices*, and the *Junos OS Feature Guides*.

Related Documentation

- *Example: Grouping Source and Destination Prefixes into a Forwarding Class*
- Configuring SCU or DCU on page 181
- Configuring SCU on a Virtual Loopback Tunnel Interface on page 184
- Configuring Class Usage Profiles on page 185
- Configuring the MIB Profile on page 187
- Configuring the Routing Engine Profile on page 189

PART 2

Configuration

CHAPTER 6

SNMP Configuration

Configuration Statements at the [edit snmp] Hierarchy Level

This topic shows all possible configuration statements at the **[edit snmp]** hierarchy level and their level in the configuration hierarchy. When you are configuring Junos OS, your current hierarchy level is shown in the banner on the line preceding the **user@host#** prompt.

```
[edit]
snmp {
  client-list client-list-name {
    ip-addresses;
  }
  community community-name {
    authorization authorization;
    client-list-name client-list-name;
    clients {
      address <restrict>;
    }
    logical-system (SNMP logical-system-name {
      routing-instance routing-instance-name;
      clients {
        address <restrict>;
      }
    }
    routing-instance routing-instance-name {
      clients {
        address <restrict>;
      }
    }
    view view-name;
  }
  contact contact;
  description description;
  engine-id {
    (local engine-id | use-default-ip-address | use-mac-address);
  }
  filter-duplicates;
  interface [ interface-names ];
  location location;
  name name;
  nonvolatile {
    commit-delay seconds;
  }
  rmon {
    alarm index {
      description description;
      falling-event-index index;
      falling-threshold integer;
      falling-threshold-interval seconds;
      interval seconds;
      request-type (get-next-request | get-request | walk-request);
      rising-event-index index;
      rising-threshold integer;
      sample-type type;
      startup-alarm alarm;
      syslog-subtag syslog-subtag;
      variable oid-variable;
    }
    event index {
      community community-name;
      description description;
      type type;
    }
```

```
}
traceoptions {
  file filename <files number> <size size> <world-readable | no-world-readable> <match
    regular-expression>;
  flag flag;
}
trap-group group-name {
  categories {
    category;
  }
  destination-port port-number;
  routing-instance instance;
  logical-system (SNMP logical-system-name;
  targets {
    address;
  }
  version (all | v1 | v2);
}
trap-options {
  agent-address outgoing-interface;
  source-address address;
  enterprise-oid;
  logical-system (SNMP logical-system-name {
    routing-instance routing-instance-name {
      source-address address;
    }
  }
  routing-instance (SNMP) routing-instance-name {
    source-address address;
  }
}
v3 {
  notify name {
    tag tag-name;
    type (trap | inform);
  }
  notify-filter profile-name {
    oid oid (include | exclude);
  }
  snmp-community community-index {
    community-name community-name;
    security-name security-name;
    tag tag-name;
  }
  target-address target-address-name {
    address address;
    address-mask address-mask;
    logical-system (SNMP logical-system;
    port port-number;
    retry-count number;
    routing-instance instance;
    tag-list tag-list;
    target-parameters target-parameters-name;
    timeout seconds;
  }
  target-parameters target-parameters-name {
```

```
    notify-filter profile-name;
    parameters {
      message-processing-model (v1 | v2c | v3);
      security-level (authentication | none | privacy);
      security-model (usm | v1 | v2c);
      security-name security-name;
    }
  }
  usm {
    local-engine {
      user username {
        authentication-md5 {
          authentication-password authentication-password;
        }
        authentication-none;
        authentication-sha {
          authentication-password authentication-password;
        }
        privacy-3des {
          privacy-password privacy-password;
        }
        privacy-aes128 {
          privacy-password privacy-password;
        }
        privacy-des {
          privacy-password privacy-password;
        }
        privacy-none;
      }
    }
  }
  vacm {
    access {
      group group-name {
        (default-context-prefix | context-prefix context-prefiix){
          security-model (any | usm | v1 | v2c) {
            security-level (authentication | none | privacy) {
              notify-view view-name;
              read-view view-name;
              write-view view-name;
            }
          }
        }
      }
    }
    security-to-group {
      security-model (usm | v1 | v2c) {
        security-name security-name {
          group group-name;
        }
      }
    }
  }
}
view view-name {
  oid object-identifier (include | exclude);
```

```
    }
}
```

Related Documentation

- Understanding the SNMP Implementation in Junos OS on page 6
- Configuring SNMP on a Device Running Junos OS on page 91

Configuring SNMP on a Device Running Junos OS

By default, SNMP is disabled on devices running Junos OS. To enable SNMP on a router or switch, you must include the SNMP configuration statements at the **[edit snmp]** hierarchy level.

To configure the minimum requirements for SNMP, include the following statements at the **[edit snmp]** hierarchy level of the configuration:

```
[edit]
snmp {
    community public;
}
```

The community defined here as **public** grants read access to all MIB data to any client.

To configure complete SNMP features, include the following statements at the **[edit snmp]** hierarchy level:

```
snmp {
    client-list client-list-name {
        ip-addresses;
    }
    community community-name {
        authorization authorization;
        client-list-name client-list-name;
        clients {
            address restrict;
        }
        routing-instance routing-instance-name {
            clients {
            addresses;
            }
        }
        logical-system logical-system-name {
            routing-instance routing-instance-name {
                clients {
                addresses;
                }
            }
        }
        view view-name;
    }
    contact contact;
    description description;
    engine-id {
        (local engine-id | use-mac-address | use-default-ip-address);
    }
```

```
    filter-duplicates;
    health-monitor {
      falling-threshold integer;
      interval seconds;
      rising-threshold integer;
    }
    interface [ interface-names ];
    location location;
    name name;
    nonvolatile {
      commit-delay seconds;
    }
    rmon {
      alarm index {
        description text-description;
        falling-event-index index;
        falling-threshold integer;
        falling-threshold-interval seconds;
        interval seconds;
        request-type (get-next-request | get-request | walk-request);
        rising-event-index index;
        sample-type type;
        startup-alarm alarm;
        syslog-subtag syslog-subtag;
        variable oid-variable;
      }
      event index {
        community community-name;
        description text-description;
        type type;
      }
    }
    traceoptions {
      file filename <files number> <size size> <world-readable | no-world-readable> <match
        regular-expression>;
      flag flag;
    }
    trap-group group-name {
      categories {
        category;
      }
      destination-port port-number;
      routing-instance instance;
      targets {
        address;
      }
      version (all | v1 | v2);
    }
    trap-options {
      agent-address outgoing-interface;
      source-address address;
    }
    view view-name {
      oid object-identifier (include | exclude);
    }
  }
```

Related Documentation
- Understanding the SNMP Implementation in Junos OS on page 6
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Complete SNMPv3 Configuration Statements on page 114

Configuring the System Contact on a Device Running Junos OS

You can specify an administrative contact for each system being managed by SNMP. This name is placed into the MIB II **sysContact** object. To configure a contact name, include the **contact** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
contact contact;
```

If the name contains spaces, enclose it in quotation marks (" ").

To define a system contact name that contains spaces:

```
[edit]
snmp {
   contact "Juniper Berry, (650) 555-1234";
}
```

Related Documentation
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuring the System Location for a Device Running Junos OS on page 93
- Configuring the System Description on a Device Running Junos OS on page 94
- Configuring the System Name on page 94
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the System Location for a Device Running Junos OS

You can specify the location of each system being managed by SNMP. This string is placed into the MIB II **sysLocation** object. To configure a system location, include the **location** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
location location;
```

If the location contains spaces, enclose it in quotation marks (" ").

To specify the system location:

```
[edit]
snmp {
   location "Row 11, Rack C";
}
```

Related Documentation
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuring the System Contact on a Device Running Junos OS on page 93
- Configuring the System Description on a Device Running Junos OS on page 94

- Configuring the System Name on page 94
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the System Description on a Device Running Junos OS

You can specify a description for each system being managed by SNMP. This string is placed into the MIB II **sysDescription** object. To configure a description, include the **description** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
description description;
```

If the description contains spaces, enclose it in quotation marks (" ").

To specify the system description:

```
[edit]
snmp {
    description "M40 router with 8 FPCs";
}
```

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuring the System Contact on a Device Running Junos OS on page 93
- Configuring the System Location for a Device Running Junos OS on page 93
- Configuring the System Name on page 94
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the System Name

Junos OS enables you to override the system name by including the **name** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
name name;
```

If the name contains spaces, enclose it in quotation marks (" ").

To specify the system name override:

```
[edit]
snmp {
    name "snmp 1";
}
```

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuring the System Contact on a Device Running Junos OS on page 93
- Configuring the System Location for a Device Running Junos OS on page 93
- Configuring the System Description on a Device Running Junos OS on page 94

- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the SNMP Community String

The SNMP community string defines the relationship between an SNMP server system and the client systems. This string acts like a password to control the clients' access to the server. To configure a community string in a Junos OS configuration, include the **community** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
community name {
    authorization authorization;
    clients {
        default restrict;
        address restrict;
    }
    view view-name;
}
```

If the community name contains spaces, enclose it in quotation marks (" ").

The default authorization level for a community is **read-only**. To allow **Set** requests within a community, you need to define that community as **authorization read-write**. For **Set** requests, you also need to include the specific MIB objects that are accessible with read-write privileges using the **view** statement. The default view includes all supported MIB objects that are accessible with read-only privileges; no MIB objects are accessible with read-write privileges. For more information about the **view** statement, see "Configuring MIB Views" on page 109.

The **clients** statement lists the IP addresses of the clients (community members) that are allowed to use this community. If no **clients** statement is present, all clients are allowed. For ***address***, you must specify an IPv4 or IPv6 address, not a hostname. Include the **default restrict** option to deny access to all SNMP clients for which access is not explicitly granted. We recommend that you always include the **default restrict** option to limit SNMP client access to the local router.

NOTE: Community names must be unique. You cannot configure the same community name at the **[edit snmp community]** and **[edit snmp v3 snmp-community *community-index*]** hierarchy levels.

Related Documentation

- Adding a Group of Clients to an SNMP Community on page 96
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Examples: Configuring the SNMP Community String on page 96

Examples: Configuring the SNMP Community String

Grant read-only access to all clients. With the following configuration, the system responds to SNMP **Get**, **GetNext**, and **GetBulk** requests that contain the community string **public**:

```
[edit]
snmp {
  community public {
    authorization read-only;
  }
}
```

Grant all clients read-write access to the ping MIB and **jnxPingMIB**. With the following configuration, the system responds to SNMP **Get**, **GetNext**, **GetBulk**, and **Set** requests that contain the community string **private** and specify an OID contained in the ping MIB or **jnxPingMIB** hierarchy:

```
[edit]
snmp {
  view ping-mib-view {
    oid pingMIB include;
    oid jnxPingMIB include;
    community private {
      authorization read-write;
      view ping-mib-view;
    }
  }
}
```

The following configuration allows read-only access to clients with IP addresses in the range **1.2.3.4/24**, and denies access to systems in the range **fe80::1:2:3:4/64**:

```
[edit]
snmp {
  community field-service {
    authorization read-only;
    clients {
      default restrict; # Restrict access to all SNMP clients not explicitly
      # listed on the following lines.
      1.2.3.4/24; # Allow access by all clients in 1.2.3.4/24 except
      fe80::1:2:3:4/64 restrict;# fe80::1:2:3:4/64.
    }
  }
}
```

Related Documentation

- Configuring the SNMP Community String on page 95

Adding a Group of Clients to an SNMP Community

Junos OS enables you to add one or more groups of clients to an SNMP community. You can include the **client-list-name *name*** statement at the **[edit snmp community *community-name*]** hierarchy level to add all the members of the client list or prefix list to an SNMP community.

To define a list of clients, include the **client-list** statement followed by the IP addresses of the clients at the **[edit snmp]** hierarchy level:

```
[edit snmp]
  client-list client-list-name {
  ip-addresses;
}
```

You can configure a prefix list at the **[edit policy options]** hierarchy level. Support for prefix lists in the SNMP community configuration enables you to use a single list to configure the SNMP and routing policies. For more information about the **prefix-list** statement, see the *Routing Policy Feature Guide for Routing Devices*.

To add a client list or prefix list to an SNMP community, include the **client-list-name** statement at the **[edit snmp community *community-name*]** hierarchy level:

```
[edit snmp community community-name]
client-list-name client-list-name;
```

NOTE: The client list and prefix list must not have the same name.

The following example shows how to define a client list:

```
[edit]
snmp {
  client-list clentlist1 {
    10.1.1.1/32;
    10.2.2.2/32;
  }
}
```

The following example shows how to add a client list to an SNMP community:

```
[edit]
snmp {
  community community1 {
    authorization read-only;
    client-list-name clientlist1;
  }
}
```

The following example shows how to add a prefix list to an SNMP community:

```
[edit]
policy-options {
  prefix-list prefixlist {
  10.3.3.3/32;
  10.5.5.5/32;
  }
}
snmp {
  community community2 {
  client-list-name prefixlist;
  }
}
```

Related Documentation

- *client-list*
- *client-list-name*

Configuring a Proxy SNMP Agent

Starting with Release 12.3, Junos OS enables you to assign one of the devices in the network as a proxy SNMP agent through which the network management system (NMS) can query other devices in the network. When you configure a proxy, you can specify the names of devices to be managed through the proxy SNMP agent.

When the NMS queries the proxy SNMP agent, the NMS specifies the community name (for SNMPv1 and SNMPv2) or the context and security name (for SNMPv3) associated with the device from which it requires the information.

NOTE: If you have configured authentication and privacy methods and passwords for SNMPv3, those parameters are also specified in the query for SNMPv3 information.

To configure a proxy SNMP agent and specify devices to be managed by the proxy SNMP agent, you can include the following configuration statements at the [**edit snmp**] hierarchy level:

```
proxy proxy-name{
  device-name device-name;
  logical-system logical-system {
    routing-instance routing-instance;
  }
  routing-instance routing-instance;
  <version-v1 | version-v2c> {
    snmp-community community-name;
    no-default-comm-to-v3-config;
  }
  version-v3 {
    security-name security-name;
    context context-name;
  }
}
```

- The **proxy** statement enables you to specify a unique name for the proxy configuration.
- The **version-v1**, **version-v2c**, and **version-v3** statements enable you to specify the SNMP version.
- The **no-default-comm-to-v3-config** statement is an optional statement at the [**edit snmp proxy *proxy-name* <version-v1 | version-v2c>**] hierarchy level that when included in the configuration requires you to manually configure the statements at the [**edit snmp v3 snmp-community *community-name***] and [**edit snmp v3 vacm**] hierarchy levels.

  If the **no-default-comm-to-v3-config** statement is not included at the [**edit snmp proxy *proxy-name* <version-v1 | version-v2c>**] hierarchy level, the [**edit snmp v3**

snmp-community *community-name*] and [**edit snmp v3 vacm**] hierarchy level configurations are automatically initialized.

- The **logical-system** and **routing-instance** statements are optional statements that enable you to specify logical system and routing instance names if you want to create proxies for logical systems or routing instances on the device.

NOTE: The community and security configuration for the proxy should match the corresponding configuration on the device that is to be managed.

NOTE: Because the proxy SNMP agent does not have trap forwarding capabilities, the devices that are managed by the proxy SNMP agent send the traps directly to the network management system.

You can use the **show snmp proxy** operational mode command to view proxy details on a device. The **show snmp proxy** command returns the proxy names, device names, SNMP version, community/security, and context information.

Related Documentation

- proxy (snmp) on page 207

Filtering Duplicate SNMP Requests

By default, filtering duplicate **get**, **getNext**, and **getBulk** SNMP requests is disabled on devices running Junos OS. If a network management station retransmits a **Get**, **GetNext**, or **GetBulk** SNMP request too frequently to the router, that request might interfere with the processing of previous requests and slow down the response time of the agent. Filtering these duplicate requests improves the response time of the SNMP agent. Junos OS uses the following information to determine if an SNMP request is a duplicate:

- Source IP address of the SNMP request
- Source UDP port of the SNMP request
- Request ID of the SNMP request

To filter duplicate SNMP requests, include the **filter-duplicates** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
filter-duplicates;
```

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuring the Interfaces on Which SNMP Requests Can Be Accepted on page 107
- Filtering Interface Information Out of SNMP Get and GetNext Output on page 108
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the Commit Delay Timer

When a router or switch first receives an SNMP nonvolatile **Set** request, a Junos OS XML protocol session opens and prevents other users or applications from changing the candidate configuration (equivalent to the command-line interface [CLI] **configure exclusive** command). If the router does not receive new SNMP **Set** requests within 5 seconds (the default value), the candidate configuration is committed and the Junos OS XML protocol session closes (the configuration lock is released). If the router receives new SNMP **Set** requests while the candidate configuration is being committed, the SNMP **Set** request is rejected and an error is generated. If the router receives new SNMP **Set** requests before 5 seconds have elapsed, the commit-delay timer (the length of time between when the last SNMP request is received and the commit is requested) resets to 5 seconds.

By default, the timer is set to 5 seconds. To configure the timer for the SNMP **Set** reply and start of the commit, include the **commit-delay** statement at the **[edit snmp nonvolatile]** hierarchy level:

```
[edit snmp nonvolatile]
commit-delay seconds;
```

seconds is the length of the time between when the SNMP request is received and the commit is requested for the candidate configuration. For more information about the **configure exclusive** command and locking the configuration, see the *CLI User Guide*.

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring SNMP Trap Options and Groups on a Device Running Junos OS

Some carriers have more than one trap receiver that forwards traps to a central NMS. This allows for more than one path for SNMP traps from a router to the central NMS through different trap receivers. A device running Junos OS can be configured to send the same copy of each SNMP trap to every trap receiver configured in the trap group.

The source address in the IP header of each SNMP trap packet is set to the address of the outgoing interface by default. When a trap receiver forwards the packet to the central NMS, the source address is preserved. The central NMS, looking only at the source address of each SNMP trap packet, assumes that each SNMP trap came from a different source.

In reality, the SNMP traps came from the same router, but each left the router through a different outgoing interface.

The statements discussed in the following sections are provided to allow the NMS to recognize the duplicate traps and to distinguish SNMPv1 traps based on the outgoing interface.

To configure SNMP trap options and trap groups, include the **trap-options** and **trap-group** statements at the **[edit snmp]** hierarchy level:

```
[edit snmp]
trap-options {
  agent-address outgoing-interface;
  source-address address;
}
trap-group group-name {
  categories {
    category;
  }
  destination-port port-number;
  targets {
    address;
  }
  version (all | v1 | v2);
}
```

Related Documentation

- Configuring SNMP Trap Options on page 101
- Configuring SNMP Trap Groups on page 104
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring SNMP Trap Options

Using SNMP trap options, you can set the source address of every SNMP trap packet sent by the router to a single address regardless of the outgoing interface. In addition, you can set the agent address of the SNMPv1 traps. For more information about the contents of SNMPv1 traps, see RFC 1157.

NOTE: SNMP cannot be associated with any routing instances other than the master routing instance.

To configure SNMP trap options, include the **trap-options** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
trap-options {
  agent-address outgoing-interface;
  enterprise-oid
  logical-system (SNMP
  routing-instance
  source-address address;
}
```

You must also configure a trap group for the trap options to take effect. For information about trap groups, see "Configuring SNMP Trap Groups" on page 104.

This topic contains the following sections:

Configuring the Source Address for SNMP Traps

You can configure the source address of trap packets in many ways: **lo0**, a valid IPv4 address configured on one of the router interfaces, a logical-system address, or the address of a routing-instance. The value **lo0** indicates that the source address of the SNMP trap packets is set to the lowest loopback address configured on the interface **lo0**.

You can configure the source address of trap packets in one of the following formats:

- a valid IPv4 address configured on one of the router interfaces
- **lo0**; that is the lowest loopback address configured on the interface **lo0**.
- a logical-system name
- a routing-instance name

A valid IPv4 Address As the Source Address

To specify a valid interface address as the source address for SNMP traps on one of the router interfaces, include the **source-address** statement at the **[edit snmp trap-options]** hierarchy level:

```
[edit snmp trap-options]
source-address address;
```

address is a valid IPv4 address configured on one of the router interfaces.

The Lowest Loopback Address As the Source Address

To specify the source address of the SNMP traps so that they use the lowest loopback address configured on the interface **lo0** as the source address, include the **source-address** statement at the **[edit snmp trap-options]** hierarchy level:

```
[edit snmp trap-options]
source-address lo0;
```

To enable and configure the loopback address, include the **address** statement at the **[edit interfaces lo0 unit 0 family inet]** hierarchy level:

```
[edit interfaces]
lo0 {
   unit 0 {
      family inet {
         address ip-address;
      }
   }
}
```

To configure the loopback address as the source address of trap packets:

```
[edit snmp]
trap-options {
```

```
    source-address lo0;
}
trap-group "urgent-dispatcher" {
    version v2;
    categories link startup;
    targets {
        192.168.10.22;
        172.17.1.2;
    }
}
[edit interfaces]
lo0 {
    unit 0 {
        family inet {
            address 10.0.0.1/32;
            address 127.0.0.1/32;
        }
    }
}
```

In this example, the IP address **10.0.0.1** is the source address of every trap sent from this router.

Logical System Name as the Source Address

To specify a logical system name as the source address of SNMP traps, include the **logical-system *logical-system-name*** statement at the **[edit snmp trap-options]** hierarchy level.

For example, the following configuration sets logical system name **ls1** as the source address of SNMP traps:

```
[edit snmp]
    trap-options{
    logical-system (SNMP ls1;
}
```

Routing Instance Name as the Source Address

To specify a routing instance name as the source address of SNMP traps, include the **routing-instance *routing-instance-name*** statement at the **[edit snmp trap-options]** hierarchy level.

For example, the following configuration sets the routing instance name **ri1** as the source address for SNMP traps:

```
[edit snmp]
    trap-options {
    routing-instance ri1;
}
```

Configuring the Agent Address for SNMP Traps

The agent address is only available in SNMPv1 trap packets (see RFC 1157). By default, the router's default local address is not specified in the agent address field of the SNMPv1 trap. To configure the agent address, include the **agent-address** statement at the **[edit snmp trap-options]** hierarchy level. Currently, the agent address can only be the address of the outgoing interface:

```
[edit snmp]
trap-options {
```

```
    agent-address outgoing-interface;
}
```

To configure the outgoing interface as the agent address:

```
[edit snmp]
trap-options {
    agent-address outgoing-interface;
}
trap-group " urgent-dispatcher" {
    version v1;
    categories link startup;
    targets {
        192.168.10.22;
        172.17.1.2;
    }
}
```

In this example, each SNMPv1 trap packet sent has its agent address value set to the IP address of the outgoing interface.

Adding snmpTrapEnterprise Object Identifier to Standard SNMP Traps

The **snmpTrapEnterprise** object helps you identify the enterprise that has defined the trap. Typically, the **snmpTrapEnterprise** object appears as the last varbind in enterprise-specific SNMP version 2 traps. However, starting Release 10.0, Junos OS enables you to add the **snmpTrapEnterprise** object identifier to standard SNMP traps as well.

To add **snmpTrapEnterprise** to standard traps, include the **enterprise-oid** statement at the **[edit snmp trap-options]** hierarchy level. If the **enterprise-oid** statement is not included in the configuration, **snmpTrapEnterprise** is added only for enterprise-specific traps.

```
[edit snmp]
trap-options {
    enterprise-oid;
}
```

Related Documentation

- Configuring SNMP Trap Options and Groups on a Device Running Junos OS on page 100
- Configuring SNMP Trap Groups on page 104
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring SNMP Trap Groups

You can create and name a group of one or more types of SNMP traps and then define which systems receive the group of SNMP traps. The trap group must be configured for SNMP traps to be sent. To create an SNMP trap group, include the **trap-group** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
trap-group group-name {
    categories {
```

```
    category;
  }
  destination-port port-number;
  routing-instance instance;
  targets {
    address;
  }
  version (all | v1 | v2);
}
```

The trap group name can be any string and is embedded in the community name field of the trap. To configure your own trap group port, include the **destination-port** statement. The default destination port is port 162.

For each trap group that you define, you must include the **target** statement to define at least one system as the recipient of the SNMP traps in the trap group. Specify the IPv4 or IPv6 address of each recipient, not its hostname.

Specify the types of traps the trap group can receive in the **categories** statement. For information about the category to which the traps belong, see the "Standard SNMP Traps Supported on Devices Running Junos OS" on page 63 and "Juniper Networks Enterprise-Specific SNMP Traps" on page 63 topics in the *SNMP MIBs and Traps Reference*.

Specify the routing instance used by the trap group in the **routing-instance** statement. All targets configured in the trap group use this routing instance.

A trap group can receive the following categories:

- **authentication**—Authentication failures
- **chassis**—Chassis or environment notifications
- **configuration**—Configuration notifications
- **link**—Link-related notifications (up-down transitions, DS-3 and DS-1 line status change, IPv6 interface state change, and Passive Monitoring PIC overload)

  NOTE: To send Passive Monitoring PIC overload interface traps, select the link trap category.

- **remote-operations**—Remote operation notifications
- **rmon-alarm**—Alarm for RMON events
- **routing**—Routing protocol notifications
- **sonet-alarms**—SONET/SDH alarms

  NOTE: If you omit the SONET/SDH subcategories, all SONET/SDH trap alarm types are included in trap notifications.

- **loss-of-light**—Loss of light alarm notification
- **pll-lock**—PLL lock alarm notification
- **loss-of-frame**—Loss of frame alarm notification
- **loss-of-signal**—Loss of signal alarm notification
- **severely-errored-frame**—Severely errored frame alarm notification
- **line-ais**—Line alarm indication signal (AIS) alarm notification
- **path-ais**—Path AIS alarm notification
- **loss-of-pointer**—Loss of pointer alarm notification
- **ber-defect**—SONET/SDH bit error rate alarm defect notification
- **ber-fault**—SONET/SDH error rate alarm fault notification
- **line-remote-defect-indication**—Line remote defect indication alarm notification
- **path-remote-defect-indication**—Path remote defect indication alarm notification
- **remote-error-indication**—Remote error indication alarm notification
- **unequipped**—Unequipped alarm notification
- **path-mismatch**—Path mismatch alarm notification
- **loss-of-cell**—Loss of cell delineation alarm notification
- **vt-ais**—Virtual tributary (VT) AIS alarm notification
- **vt-loss-of-pointer**—VT loss of pointer alarm notification
- **vt-remote-defect-indication**—VT remote defect indication alarm notification
- **vt-unequipped**—VT unequipped alarm notification
- **vt-label-mismatch**—VT label mismatch error notification
- **vt-loss-of-cell**—VT loss of cell delineation notification

- **startup**—System warm and cold starts
- **vrrp-events**—Virtual Router Redundancy Protocol (VRRP) events such as new-master or authentication failures

To enable the **authenticationFailure** SNMP trap, you can configure either the **startup** or the **authentication** type of SNMP traps for the trap group:

- **startup** type of SNMP traps:

```
trap-group customer {
  categories {
    startup;
  }
}
```

- **authentication** type of SNMP traps:

```
trap-group customer {
```

```
    categories {
      authentication;
    }
  }
```

If you include SONET/SDH subcategories, only those SONET/SDH trap alarm types are included in trap notifications.

The **version** statement allows you to specify the SNMP version of the traps sent to targets of the trap group. If you specify **v1** only, SNMPv1 traps are sent. If you specify **v2** only, SNMPv2 traps are sent. If you specify **all**, both an SNMPv1 and an SNMPv2 trap are sent for every trap condition. For more information about the **version** statement, see version (SNMP).

Related Documentation

- Configuring SNMP Trap Options and Groups on a Device Running Junos OS on page 100
- Configuring SNMP Trap Options on page 101
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Example: Configuring SNMP Trap Groups on page 107

Example: Configuring SNMP Trap Groups

Set up a trap notification list named **urgent-dispatcher** for link and startup traps. This list is used to identify the network management hosts (**1.2.3.4** and **fe80::1:2:3:4**) to which traps generated by the local router should be sent. The name specified for a trap group is used as the SNMP community string when the agent sends traps to the listed targets.

```
[edit]
snmp {
  trap-group "urgent-dispatcher" {
    version v2;
    categories link startup;
    targets {
      1.2.3.4;
      fe80::1:2:3:4;
    }
  }
}
```

Related Documentation

- Configuring SNMP Trap Groups on page 104
- Configuring SNMP Trap Options and Groups on a Device Running Junos OS on page 100
- Configuring SNMP Trap Options on page 101

Configuring the Interfaces on Which SNMP Requests Can Be Accepted

By default, all router or switch interfaces have SNMP access privileges. To limit the access through certain interfaces only, include the **interface** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
interface [ interface-names ];
```

Specify the names of any logical or physical interfaces that should have SNMP access privileges. Any SNMP requests entering the router or switch from interfaces not listed are discarded.

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Example: Configuring Secured Access List Checking on page 108
- *Configuring SNMP*

Example: Configuring Secured Access List Checking

Grant SNMP access privileges only to devices on interfaces **so-0/0/0** and **at-1/0/1**. The following example does this by configuring a list of logical interfaces:

```
[edit]
snmp {
  interface [ so-0/0/0.0 so-0/0/0.1 at-1/0/1.0 at-1/0/1.1 ];
}
```

The following example grants the same access by configuring a list of physical interfaces:

```
[edit]
snmp {
  interface [ so-0/0/0 at-1/0/1 ];
}
```

Related Documentation

- Configuring the Interfaces on Which SNMP Requests Can Be Accepted on page 107
- Filtering Interface Information Out of SNMP Get and GetNext Output on page 108
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Filtering Interface Information Out of SNMP Get and GetNext Output

Junos OS enables you to filter out information related to specific interfaces from the output of SNMP **Get** and **GetNext** requests performed on interface-related MIBs such as IF MIB, ATM MIB, RMON MIB, and the Juniper Networks enterprise-specific IF MIB.

You can use the following options of the **filter-interfaces** statement at the **[edit snmp]** hierarchy level to specify the interfaces that you want to exclude from SNMP **Get** and **GetNext** queries:

- **interfaces**—Interfaces that match the specified regular expressions.
- **all-internal-interfaces**—Internal interfaces.

```
[edit]
```

```
    snmp {
    filter-interfaces {
      interfaces {
        interface1;
        interface2;
      }
      all-internal-interfaces;
    }
}
```

Starting with Release 12.1, Junos OS provides an except option (**!** operator) that enables you to filter out all interfaces except those interfaces that match all the regular expressions prefixed with the **!** mark.

For example, to filter out all interfaces except the **ge** interfaces from the SNMP **get** and **get-next** results, enter the following command:

```
[edit snmp]
user@host# set filter-interfaces interfaces "!^~ge-.*"
user@host# commit
```

When this is configured, Junos OS filters out all interfaces except the **ge** interfaces from the SNMP **get** and **get-next** results.

NOTE: The **!** mark is supported only as the first character of the regular expression. If it appears anywhere else in a regular expression, Junos OS considers the regular expression invalid, and returns an error.

However, note that these settings are limited to SNMP operations, and the users can continue to access information related to the interfaces (including those hidden using the **filter-interfaces** options) using the appropriate Junos OS command-line interface (CLI) commands.

Related Documentation

- Configuring the Interfaces on Which SNMP Requests Can Be Accepted on page 107
- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring MIB Views

By default, an SNMP community grants read access and denies write access to all supported MIB objects (even communities configured as **authorization read-write**). To restrict or grant read or write access to a set of MIB objects, you must configure a MIB view and associate the view with a community.

To configure MIB views, include the **view** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
view view-name {
  oid object-identifier (include | exclude);
}
```

The **view** statement defines a MIB view and identifies a group of MIB objects. Each MIB object of a view has a common object identifier (OID) prefix. Each object identifier represents a subtree of the MIB object hierarchy. The subtree can be represented either by a sequence of dotted integers (such as **1.3.6.1.2.1.2**) or by its subtree name (such as **interfaces**). A configuration statement uses a view to specify a group of MIB objects on which to define access. You can also use a wildcard character asterisk (*) to include OIDs that match a particular pattern in the SNMP view. To enable a view, you must associate the view with a community.

NOTE: To remove an OID completely, use the **delete view all oid *oid-number*** command but omit the **include** parameter.

To associate MIB views with a community, include the **view** statement at the **[edit snmp community *community-name*]** hierarchy level:

```
[edit snmp community community-name]
view view-name;
```

For more information about the Ping MIB, see RFC 2925 and the *PING MIB* topic in the *SNMP MIBs and Traps Reference*.

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Example: Ping Proxy MIB on page 110
- **view (Configuring a MIB View) on page 216**
- *view (Associating MIB View with a Community)*
- **oid on page 206**

Example: Ping Proxy MIB

Restrict the ***ping-mib*** community to read and write access of the Ping MIB and **jnxpingMIB** only. Read or write access to any other MIB using this community is not allowed.

```
[edit snmp]
view ping-mib-view {
   oid 1.3.6.1.2.1.80 include; #pingMIB
   oid jnxPingMIB include; #jnxPingMIB
}
community ping-mib {
   authorization read-write;
   view ping-mib-view;
}
```

The following configuration prevents the ***no-ping-mib*** community from accessing Ping MIB and **jnxPingMIB** objects. However, this configuration does not prevent the ***no-ping-mib*** community from accessing any other MIB object that is supported on the device.

```
[edit snmp]
view no-ping-mib-view {
```

```
    oid 1.3.6.1.2.1.80 exclude; # deny access to pingMIB objects
    oid jnxPingMIB exclude; # deny access to jnxPingMIB objects
}
community no-ping-mib {
    authorization read-write;
    view ping-mib-view;
}
```

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Configuring MIB Views on page 109
- **view (Configuring a MIB View) on page 216**
- **oid on page 206**

Example: Tracing SNMP Activity

Trace information about SNMP packets:

```
[edit]
snmp {
    traceoptions {
        file size 10k files 5;
        flag pdu;
        flag protocol-timeouts;
        flag varbind-error;
    }
}
```

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Tracing SNMP Activity on a Device Running Junos OS on page 299
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87

Configuring the Local Engine ID

By default, the local engine ID uses the default IP address of the router. The local engine ID is the administratively unique identifier for the SNMPv3 engine. This statement is optional. To configure the local engine ID, include the **engine-id** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
engine-id {
    (local engine-id-suffix | use-default-ip-address | use-mac-address);
}
```

- **local *engine-id-suffix***—The engine ID suffix is explicitly configured.
- **use-default-ip-address**—The engine ID suffix is generated from the default IP address.
- **use-mac-address**—The SNMP engine identifier is generated from the Media Access Control (MAC) address of the management interface on the router.

The local engine ID is defined as the administratively unique identifier of an SNMPv3 engine, and is used for identification, not for addressing. There are two parts of an engine ID: prefix and suffix. The prefix is formatted according to the specifications defined in RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*. You can configure the suffix here.

NOTE: SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID. If you configure or change the engine ID, you must commit the new engine ID before you configure SNMPv3 users. Otherwise the keys generated from the configured passwords are based on the previous engine ID. For the engine ID, we recommend using the master IP address of the device if the device has multiple routing engines and has the master IP address configured. Alternatively, you can use the MAC address of the management port if the device has only one Routing Engine.

Related Documentation

- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: SNMPv3 Configuration on page 147

CHAPTER 7

SNMPv3

Complete SNMPv3 Configuration Statements

To configure SNMPv3, include the following statements at the **[edit snmp v3]** and **[edit snmp]** hierarchy levels:

```
[edit snmp]
engine-id {
    (local engine-id | use-mac-address | use-default-ip-address);
}
view view-name {
    oid object-identifier (include | exclude);
}
[edit snmp v3]
notify name {
    tag tag-name;
    type (trap | inform);
}
notify-filter profile-name {
    oid object-identifier (include | exclude);
}
snmp-community community-index {
    community-name community-name;
    security-name security-name;
    tag tag-name;
}
target-address target-address-name {
    address address;
    address-mask address-mask;
    logical-system (SNMP logical-system;
    port port-number;
    retry-count number;
    routing-instance instance;
    tag-list tag-list;
    target-parameters target-parameters-name;
    timeout seconds;
}
target-parameters target-parameters-name {
    notify-filter profile-name;
    parameters {
        message-processing-model (v1 | v2c | v3);
        security-level (authentication | none | privacy);
        security-model (usm | v1 | v2c);
        security-name security-name;
    }
}
usm {
    (local-engine | remote-engine engine-id) {
        user username {
            authentication-md5 {
                authentication-password authentication-password;
            }
            authentication-none;
            authentication-sha {
                authentication-password authentication-password;
```

```
        }
        privacy-3des {
          privacy-password privacy-password;
        }
        privacy-aes128 {
          privacy-password privacy-password;
        }
        privacy-des {
          privacy-password privacy-password;
        }
        privacy-none;
      }
    }
  }
  vacm {
    access {
      group group-name {
        (default-context-prefix | context-prefix context-prefix){
          security-model (any | usm | v1 | v2c) {
            security-level (authentication | none | privacy) {
              notify-view view-name;
              read-view view-name;
              write-view view-name;
            }
          }
        }
      }
    }
    security-to-group {
      security-model (usm | v1 | v2c) {
        security-name security-name {
          group group-name;
        }
      }
    }
  }
```

Related Documentation

- Creating SNMPv3 Users on page 118
- Configuring MIB Views on page 109
- Defining Access Privileges for an SNMP Group on page 122
- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring SNMP Informs on page 139
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Minimum SNMPv3 Configuration on a Device Running Junos OS

To configure the minimum requirements for SNMPv3, include the following statements at the **[edit snmp v3]** and **[edit snmp]** hierarchy levels:

NOTE: You must configure at least one view (notify, read, or write) at the **[edit snmp view-name]** hierarchy level.

```
[edit snmp]
view view-name {
  oid object-identifier (include | exclude);
}
[edit snmp v3]
notify name {
  tag tag-name;
}
notify-filter profile-name {
  oid object-identifier (include | exclude);
}
snmp-community community-index {
  security-name security-name;
}
target-address target-address-name {
  address address;
  target-parameters target-parameters-name;
}
target-parameters target-parameters-name {
  notify-filter profile-name;
  parameters {
    message-processing-model (v1 | v2c | v3);
    security-level (authentication | none | privacy);
    security-model (usm | v1 | v2c);
    security-name security-name;
  }
}
usm {
  local-engine {
    user username {
    }
  }
}
vacm {
  access {
    group group-name {
      (default-context-prefix | context-prefix context-prefix){
        security-model (any | usm | v1 | v2c) {
          security-level (authentication | none | privacy) {
            notify-view view-name;
            read-view view-name;
            write-view view-name;
          }
        }
```

```
        }
      }
    }
    security-to-group {
      security-model (usm | v1 | v2c) {
        security-name security-name {
          group group-name;
        }
      }
    }
  }
```

Related Documentation

- Creating SNMPv3 Users on page 118
- Configuring MIB Views on page 109
- Defining Access Privileges for an SNMP Group on page 122
- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring SNMP Informs on page 139
- Complete SNMPv3 Configuration Statements on page 114
- Example: SNMPv3 Configuration on page 147

Configuring the Local Engine ID

By default, the local engine ID uses the default IP address of the router. The local engine ID is the administratively unique identifier for the SNMPv3 engine. This statement is optional. To configure the local engine ID, include the **engine-id** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
engine-id {
  (local engine-id-suffix | use-default-ip-address | use-mac-address);
}
```

- **local *engine-id-suffix***—The engine ID suffix is explicitly configured.
- **use-default-ip-address**—The engine ID suffix is generated from the default IP address.
- **use-mac-address**—The SNMP engine identifier is generated from the Media Access Control (MAC) address of the management interface on the router.

The local engine ID is defined as the administratively unique identifier of an SNMPv3 engine, and is used for identification, not for addressing. There are two parts of an engine ID: prefix and suffix. The prefix is formatted according to the specifications defined in RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*. You can configure the suffix here.

NOTE: SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID. If you configure or change the engine ID, you must commit the new engine ID before you configure SNMPv3 users. Otherwise the keys generated from the configured passwords are based on the previous engine ID. For the engine ID, we recommend using the master IP address of the device if the device has multiple routing engines and has the master IP address configured. Alternatively, you can use the MAC address of the management port if the device has only one Routing Engine.

Related Documentation

- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: SNMPv3 Configuration on page 147

Creating SNMPv3 Users

For each SNMPv3 user, you can specify the username, authentication type, authentication password, privacy type, and privacy password. After a user enters a password, a key based on the engine ID and password is generated and is written to the configuration file. After the generation of the key, the password is deleted from this configuration file.

NOTE: You can configure only one encryption type for each SNMPv3 user.

To create users, include the **user** statement at the **[edit snmp v3 usm local-engine]** hierarchy level:

```
[edit snmp v3 usm local-engine]
user username;
```

username is the name that identifies the SNMPv3 user.

To configure user authentication and encryption, include the following statements at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
authentication-md5 {
    authentication-password authentication-password;
}
authentication-sha {
    authentication-password authentication-password;
}
authentication-none;
privacy-aes128 {
    privacy-password privacy-password;
}
privacy-des {
    privacy-password privacy-password;
}
privacy-3des {
```

```
    privacy-password privacy-password;
  }
  privacy-none;
```

Related Documentation

- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- *Example: Creating SNMPv3 Users Configuration*
- Example: SNMPv3 Configuration on page 147

Configuring the SNMPv3 Authentication Type

By default, in a Junos OS configuration the SNMPv3 authentication type is set to none.

This topic includes the following sections:

Configuring MD5 Authentication

To configure the message digest algorithm (MD5) as the authentication type for an SNMPv3 user, include the **authentication-md5** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
authentication-md5 {
  authentication-password authentication-password;
}
```

authentication-password is the password used to generate the key used for authentication.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Configuring SHA Authentication

To configure the secure hash algorithm (SHA) as the authentication type for an SNMPv3 user, include the **authentication-sha** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
authentication-sha {
  authentication-password authentication-password;
}
```

authentication-password is the password used to generate the key used for authentication.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Configuring No Authentication

To configure no authentication for an SNMPv3 user, include the **authentication-none** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
authentication-none;
```

Related Documentation

- Configuring the Encryption Type on page 120
- Defining Access Privileges for an SNMP Group on page 122
- Configuring the Access Privileges Granted to a Group on page 123
- Assigning Security Model and Security Name to a Group on page 127
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the Encryption Type

By default, encryption is set to none.

NOTE: Before you configure encryption, you must configure MD5 or SHA authentication.

Before you configure the privacy-des, privacy-3des and privacy-aes128 statements, you must install the jcrypto package, and either restart the SNMP process or reboot the router.

This topic includes the following sections:

- Configuring the Advanced Encryption Standard Algorithm on page 121
- Configuring the Data Encryption Algorithm on page 121
- Configuring Triple DES on page 121
- Configuring No Encryption on page 122

Configuring the Advanced Encryption Standard Algorithm

To configure the Advanced Encryption Standard (AES) algorithm for an SNMPv3 user, include the **privacy-aes128** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
privacy-aes128 {
    privacy-password privacy-password;
}
```

privacy-password is the password used to generate the key used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Configuring the Data Encryption Algorithm

To configure the data encryption algorithm (DES) for an SNMPv3 user, include the **privacy-des** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
privacy-des {
    privacy-password privacy-password;
}
```

privacy-password is the password used to generate the key used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Configuring Triple DES

To configure triple DES for an SNMPv3 user, include the **privacy-3des** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
privacy-3des {
    privacy-password privacy-password;
}
```

privacy-password is the password used to generate the key used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Configuring No Encryption

To configure no encryption for an SNMPv3 user, include the **privacy-none** statement at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level:

```
[edit snmp v3 usm local-engine user username]
privacy-none;
```

Related Documentation

- Configuring the SNMPv3 Authentication Type on page 119
- Defining Access Privileges for an SNMP Group on page 122
- Configuring the Access Privileges Granted to a Group on page 123
- Assigning Security Model and Security Name to a Group on page 127
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Defining Access Privileges for an SNMP Group

The SNMP version 3 (SNMPv3) uses the view-based access control model (VACM), which allows you to configure the access privileges granted to a group. Access is controlled by filtering the MIB objects available for a specific operation through a predefined view. You assign views to determine the objects that are visible for read, write, and notify operations for a particular group, using a particular context, a particular security model (v1, v2c, or usm), and particular security level (authenticated, privacy, or none). For information about how to configure views, see "Configuring MIB Views" on page 109.

You define user access to management information at the **[edit snmp v3 vacm]** hierarchy level. All access control within VACM operates on groups, which are collections of users as defined by USM, or community strings as defined in the SNMPv1 and SNMPv2c security models. The term ***security-name*** refers to these generic end users. The group to which a specific security name belongs is configured at the **[edit snmp v3 vacm security-to-group]** hierarchy level. That security name can be associated with a group defined at the **[edit snmp v3 vacm security-to-group]** hierarchy level. A group identifies a collection of SNMP users that share the same access policy. You then define the access privileges associated with a group at the **[edit snmp v3 vacm access]** hierarchy level. Access privileges are defined using views. For each group, you can apply different views depending on the SNMP operation; for example, read (**get**, **getNext**, or **getBulk**) write (**set**), notifications, the security level used (authentication, privacy, or none), and the security model (v1, v2c, or usm) used within an SNMP request.

You configure members of a group with the **security-name** statement. For v3 packets using USM, the security name is the same as the username. For SNMPv1 or SNMPv2c packets, the security name is determined based on the community string. Security names are specific to a security model. If you are also configuring VACM access policies for

SNMPv1 or SNMPv2c packets, you must assign security names to groups for each security model (SNMPv1 or SNMPv2c) at the **[edit snmp v3 vacm security-to-group]** hierarchy level. You must also associate a security name with an SNMP community at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level.

To configure the access privileges for an SNMP group, include statements at the **[edit snmp v3 vacm]** hierarchy level:

```
[edit snmp v3 vacm]
access {
  group group-name {
    (default-context-prefix | context-prefix context-prefix){
      security-model (any | usm | v1 | v2c) {
        security-level (authentication | none | privacy) {
          notify-view view-name;
          read-view view-name;
          write-view view-name;
        }
      }
    }
  }
}
security-to-group {
  security-model (usm | v1 | v2c) {
    security-name security-name {
      group group-name;
    }
  }
}
```

Related Documentation

- Configuring the SNMPv3 Authentication Type on page 119
- Configuring the Access Privileges Granted to a Group on page 123
- Assigning Security Model and Security Name to a Group on page 127
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the Access Privileges Granted to a Group

This topic includes the following sections:

Configuring the Group

To configure the access privileges granted to a group, include the **group** statement at the **[edit snmp v3 vacm access]** hierarchy level:

```
[edit snmp v3 vacm access]
group group-name;
```

group-name is a collection of SNMP users that belong to a common SNMP list that defines an access policy. Users belonging to a particular SNMP group inherit all access privileges granted to that group.

Configuring the Security Model

To configure the security model, include the **security-model** statement at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*)]** hierarchy level:

```
[edit snmp v3 vacm access group group-name (default-context-prefix | context-prefix
    context-prefix)]
security-model (any | usm | v1 | v2c);
```

- **any**—Any security model
- **usm**—SNMPv3 security model
- **v1**—SNMPV1 security model
- **v2c**—SNMPv2c security model

Configuring the Security Level

To configure the access privileges granted to packets with a particular security level, include the **security-level** statement at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c)]** hierarchy level:

```
[edit snmp v3 vacm access group group-name default-context-prefix security-model (any
    | usm | v1 | v2c)]
security-level (authentication | none | privacy);
```

- **none**—Provides no authentication and no encryption.
- **authentication**—Provides authentication but no encryption.
- **privacy**—Provides authentication and encryption.

NOTE: Access privileges are granted to all packets with a security level equal to or greater than that configured. If you are configuring the SNMPv1 or SNMPv2c security model, use **none** as your security level. If you are configuring the SNMPv3 security model (USM), use the **authentication**, **none**, or **privacy** security level.

Associating MIB Views with an SNMP User Group

MIB views define access privileges for members of a group. Separate views can be applied for each SNMP operation (read, write, and notify) within each security model (usm, v1, and v2c) and each security level (authentication, none, and privacy) supported by SNMP.

To associate MIB views with an SNMP user group, include the following statements at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]** hierarchy level:

```
[edit snmp v3 vacm accessgroup group-name (default-context-prefix | context-prefix
   context-prefix)security-model (any | usm | v1 | v2c) security-level (authentication | none
   | privacy)]
notify-view view-name;
read-view view-name;
write-view view-name;
```

NOTE: You must associate at least one view (notify, read, or write) at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]** hierarchy level.

You must configure the MIB view at the **[edit snmp view *view-name*]** hierarchy level. For information about how to configure MIB views, see "Configuring MIB Views" on page 109.

This section describes the following topics related to this configuration:

Configuring the Notify View

To associate notify access with an SNMP user group, include the **notify-view** statement at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]** hierarchy level:

```
[edit snmp v3 vacm access group group-name (default-context-prefix | context-prefix
   context-prefix) security-model (any | usm | v1 | v2c) security-level (authentication | none
   | privacy)]
notify-view view-name;
```

view-name specifies the notify access, which is a list of notifications that can be sent to each user in an SNMP group. A view name cannot exceed 32 characters.

Configuring the Read View

To associate a read view with an SNMP group, include the **read-view** statement at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]** hierarchy level:

```
[edit snmp v3 vacm access group group-name (default-context-prefix | context-prefix
  context-prefix) security-model (any | usm | v1 | v2c) security-level (authentication | none
  | privacy)]
read-view view-name;
```

view-name specifies read access for an SNMP user group. A view name cannot exceed 32 characters.

Configuring the Write View

To associate a write view with an SNMP user group, include the **write-view** statement at the **[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]** hierarchy level:

```
[edit snmp v3 vacm access group group-name (default-context-prefix | context-prefix
  context-prefix) security-model (any | usm | v1 | v2c) security-level (authentication | none
  | privacy)]
write-view view-name;
```

view-name specifies write access for an SNMP user group. A view name cannot exceed 32 characters.

Related Documentation

- Configuring the SNMPv3 Authentication Type on page 119
- Defining Access Privileges for an SNMP Group on page 122
- Assigning Security Model and Security Name to a Group on page 127
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: Access Privilege Configuration on page 126

Example: Access Privilege Configuration

Define access privileges:

```
[edit snmp v3]
access {
  group group1 {
    default-context-prefix {
      security-model usm {        #Define an SNMPv3 security model
        security-level privacy {
          notify-view nv1;
          read-view rv1;
          write-view wv1;
        }
```

```
        }
      }
      context-prefix lr1/ri1{ # routing instance ri1 in logical system lr1
        security-model usm {
          security-level privacy {
            notify-view nv1;
            read-view rv1;
            write-view wv1;
          }
        }
      }
    }
    group group2 {
      default-context-prefix {
        security-model usm {        #Define an SNMPv3 security model
          security-level authentication {
            read-view rv2;
            write-view wv2;
          }
        }
      }
    }
    group group3 {
      default-context-prefix {
        security-model v1 {        #Define an SNMPv3 security model
          security-level none {
            read-view rv3;
            write-view wv3;
          }
        }
      }
    }
  }
```

Related Documentation

- Configuring the Access Privileges Granted to a Group on page 123
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Assigning Security Model and Security Name to a Group

To assign security names to groups, include the following statements at the **[edit snmp v3 vacm security-to-group]** hierarchy level:

```
[edit snmp v3 vacm security-to-group]
security-model (usm | v1 | v2c) {
  security-name security-name {
    group group-name;
  }
}
```

This topic includes the following sections:

Configuring the Security Model

To configure the security model, include the **security-model** statement at the **[edit snmp v3 vacm security-to-group]** hierarchy level:

```
[edit snmp v3 vacm security-to-group]
security-model (usm | v1 | v2c);
```

- **usm**—SNMPv3 security model
- **v1**—SNMPv1 security model
- **v2c**—SNMPv2 security model

Assigning Security Names to Groups

To associate a security name with an SNMPv3 user, or a v1 or v2 community string, include the **security-name** statement at the **[edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c)]** hierarchy level:

```
[edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c)]
security-name security-name;
```

For SNMPv3, the ***security-name*** is the username configured at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level. For SNMPv1 and SNMPv2c, the security name is the community string configured at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level. For information about configuring usernames, see "Creating SNMPv3 Users" on page 118. For information about configuring a community string, see "Configuring the SNMPv3 Community" on page 144.

NOTE: The USM security name is separate from the SNMPv1 and SNMPv2c security name. If you support SNMPv1 and SNMPv2c in addition to SNMPv3, you must configure separate security names within the security-to-group configuration at the **[edit snmp v3 vacm access]** hierarchy level.

Configuring the Group

After you have created SNMPv3 users, or v1 or v2 security names, you associate them with a group. A group is a set of security names belonging to a particular security model. A group defines the access rights for all users belonging to it. Access rights define what SNMP objects can be read, written to, or created. A group also defines what notifications a user is allowed to receive.

If you already have a group that is configured with all of the view and access permissions that you want to give a user, you can add the user to that group. If you want to give a user

view and access permissions that no other groups have, or if you do not have any groups configured, create a group and add the user to it.

To configure the access privileges granted to a group, include the **group** statement at the **[edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c) security-name *security-name*]** hierarchy level:

```
[edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c) security-name
  security-name]
group group-name;
```

group-name identifies a collection of SNMP security names that share the same access policy. For more information about groups, see "Defining Access Privileges for an SNMP Group" on page 122.

Example: Security Group Configuration

Assign security names to groups:

```
vacm {
  security-to-group {
    security-model usm {
      security-name user1 {
        group group1;
      }
      security-name user2 {
        group group2;
      }
      security-name user3 {
        group group3;
      }
    }
  }
}
```

Related Documentation

- Assigning Security Model and Security Name to a Group on page 127
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring SNMPv3 Traps on a Device Running Junos OS

In SNMPv3, you create traps and informs by configuring the **notify**, **target-address**, and **target-parameters** parameters. Traps are unconfirmed notifications, whereas informs are confirmed notifications. This section describes how to configure SNMP traps. For information about configuring SNMP informs, see "Configuring SNMP Informs" on page 139.

The target address defines a management application's address and parameters to be used in sending notifications. Target parameters define the message processing and security parameters that are used in sending notifications to a particular management target. SNMPv3 also lets you define SNMPv1 and SNMPv2c traps.

NOTE: When you configure SNMP traps, make sure your configured access privileges allow the traps to be sent. Access privileges are configured at the **[edit snmp v3 vacm access]** and **[edit snmp v3 vacm security-to-group]** hierarchy levels.

To configure SNMP traps, include the following statements at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
notify name {
    tag tag-name;
    type trap;
}
notify-filter name {
    oid object-identifier (include | exclude);
}
target-address target-address-name {
    address address;
    address-mask address-mask;
    logical-system (SNMP logical-system;
    port port-number;
    routing-instance instance;
    tag-list tag-list;
    target-parameters target-parameters-name;
}
target-parameters target-parameters-name {
    notify-filter profile-name;
    parameters {
        message-processing-model (v1 | v2c | v3);
        security-level (authentication | none | privacy);
        security-model (usm | v1 | v2c);
        security-name security-name;
    }
}
```

Related Documentation

- Configuring the SNMPv3 Trap Notification on page 131
- Configuring the Trap Notification Filter on page 132
- Configuring the Trap Target Address on page 133
- Defining and Configuring the Trap Target Parameters on page 136
- Configuring SNMP Informs on page 139
- Configuring the Remote Engine and Remote User on page 140
- Configuring the Inform Notification Type and Target Address on page 142
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the SNMPv3 Trap Notification

The **notify** statement specifies the type of notification (trap) and contains a single tag. The tag defines a set of target addresses to receive a trap. The tag list contains one or more tags and is configured at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level. If the tag list contains this tag, Junos OS sends a notification to all the target addresses associated with this tag.

To configure the trap notifications, include the **notify** statement at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
notify name {
  tag tag-name;
  type trap;
}
```

name is the name assigned to the notification.

tag-name defines the target addresses to which this notification is sent. This notification is sent to all the target-addresses that have this tag in their tag list. The ***tag-name*** is not included in the notification.

trap is the type of notification.

NOTE: Each notify entry name must be unique.

Junos OS supports two types of notification: trap and inform.

For information about how to configure the tag list, see "Configuring the Trap Target Address" on page 134.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the Trap Notification Filter on page 132
- Configuring the Trap Target Address on page 133
- Defining and Configuring the Trap Target Parameters on page 136
- Configuring SNMP Informs on page 139
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: Configuring SNMPv3 Trap Notification on page 131

Example: Configuring SNMPv3 Trap Notification

Specify three sets of destinations to send traps:

```
[edit snmp v3]
```

```
notify n1 {
    tag router1;
    type trap;
}
notify n2 {
    tag router2;
    type trap
}
notify n3 {
    tag router3;
    type trap;
}
```

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the Trap Notification Filter

SNMPv3 uses the notify filter to define which traps (or which objects from which traps) are sent to the network management system (NMS). The trap notification filter limits the type of traps that are sent to the NMS.

Each object identifier represents a subtree of the MIB object hierarchy. The subtree can be represented either by a sequence of dotted integers (such as **1.3.6.1.2.1.2**) or by its subtree name (such as **interfaces**). You can also use the wildcard character asterisk (*) in the object identifier (OID) to specify object identifiers that match a particular pattern.

To configure the trap notifications filter, include the **notify-filter** statement at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
notify-filter profile-name;
```

profile-name is the name assigned to the notify filter.

By default, the OID is set to **include**. To define access to traps (or objects from traps), include the **oid** statement at the **[edit snmp v3 notify-filter *profile-name*]** hierarchy level:

```
[edit snmp v3 notify-filter profile-name]
oid oid (include | exclude);
```

oid is the object identifier. All MIB objects represented by this statement have the specified OID as a prefix. It can be specified either by a sequence of dotted integers or by a subtree name.

- **include**—Include the subtree of MIB objects represented by the specified OID.
- **exclude**—Exclude the subtree of MIB objects represented by the specified OID.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the SNMPv3 Trap Notification on page 131

Configuring the Trap Target Address

The target address defines a management application's address and parameters that are used in sending notifications. It can also identify management stations that are allowed to use specific community strings. When you receive a packet with a recognized community string and a tag is associated with it, Junos OS looks up all the target addresses with this tag and verifies that the source address of this packet matches one of the configured target addresses.

NOTE: You must configure the address mask when you configure the SNMP community.

To specify where you want the traps to be sent and define what SNMPv1 and SNMPv2cc packets are allowed, include the **target-address** statement at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
target-address target-address-name;
```

target-address-name is the string that identifies the target address.

To configure the target address properties, include the following statements at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
address address;
address-mask address-mask;
logical-system (SNMP logical-system;
port port-number;
routing-instance instance;
tag-list tag-list;
target-parameters target-parameters-name;
```

This section includes the following topics:

Configuring the Address

To configure the address, include the **address** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
address address;
```

address is the SNMP target address.

Configuring the Address Mask

The address mask specifies a set of addresses that are allowed to use a community string and verifies the source addresses for a group of target addresses.

To configure the address mask, include the **address-mask** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
address-mask address-mask;
```

address-mask combined with the address defines a range of addresses. For information about how to configure the community string, see "Configuring the SNMPv3 Community" on page 144.

Configuring the Port

By default, the UDP port is set to 162. To configure a different port number, include the **port** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
port port-number;
```

port-number is the SNMP target port number.

Configuring the Routing Instance

Traps are sent over the default routing instance. To configure the routing instance for sending traps, include the **routing-instance** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
routing-instance instance;
```

instance is the name of the routing instance. To configure a routing instance within a logical system, specify the logical system name followed by the routing instance name. Use a slash (/) to separate the two names (for example, **test-lr/test-ri**). To configure the default routing instance on a logical system, specify the logical system name followed by **default** (for example, **test-lr/default**).

Configuring the Trap Target Address

Each **target-address** statement can have one or more tags configured in its tag list. Each tag can appear in more than one tag list. When a significant event occurs on the network device, the tag list identifies the targets to which a notification is sent.

To configure the tag list, include the **tag-list** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
tag-list "tag-list";
```

tag-list specifies one or more tags as a space-separated list enclosed within double quotes.

For an example of tag list configuration, see "Example: Configuring the Tag List" on page 136.

For information about how to specify a tag at the **[edit snmp v3 notify *notify-name*]** hierarchy level, see "Configuring the SNMPv3 Trap Notification" on page 131.

NOTE: When you configure SNMP traps, make sure your configured access privileges allow the traps to be sent. Configure access privileges at the **[edit snmp v3 vacm access]** hierarchy level.

Applying Target Parameters

The **target-parameters** statement at the **[edit snmp v3]** hierarchy level applies the target parameters configured at the **[edit snmp v3 target-parameters *target-parameters-name*]** hierarchy level.

To reference configured target parameters, include the **target-parameters** statement at the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
[edit snmp v3 target-address target-address-name]
target-parameters target-parameters-name;
```

target-parameters-name is the name associated with the message processing and security parameters that are used in sending notifications to a particular management target.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the SNMPv3 Trap Notification on page 131
- Configuring the Trap Notification Filter on page 132
- Defining and Configuring the Trap Target Parameters on page 136
- Configuring SNMP Informs on page 139
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: Configuring the Tag List on page 136

Example: Configuring the Tag List

In the following example, two tag entries (**router1** and **router2**) are defined at the **[edit snmp v3 notify *notify-name*]** hierarchy level. When an event triggers a notification, Junos OS sends a trap to all target addresses that have **router1** or **router2** configured in their target-address tag list. This results in the first two targets getting one trap each, and the third target getting two traps.

```
[edit snmp v3]
notify n1 {
   tag router1; # Identifies a set of target addresses
   type trap; # Defines the type of notification
}
notify n2 {
   tag router2;
   type trap;
}
target-address ta1 {
   address 10.1.1.1;
   address-mask 255.255.255.0;
   port 162;
   tag-list router1;
   target-parameters tp1;
}
target-address ta2 {
   address 10.1.1.2;
   address-mask 255.255.255.0;
   port 162;
   tag-list router2;
   target-parameters tp2;
}
target-address ta3 {
   address 10.1.1.3;
   address-mask 255.255.255.0;
   port 162;
   tag-list “router1 router2”; #Define multiple tags in the target address tag list
   target-parameters tp3;
}
```

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the Trap Target Address on page 133
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Defining and Configuring the Trap Target Parameters

Target parameters define the message processing and security parameters that are used in sending notifications to a particular management target.

To define a set of target parameters, include the **target-parameters** statement at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
target-parameters target-parameters-name;
```

target-parameters-name is the name assigned to the target parameters.

To configure target parameter properties, include the following statements at the **[edit snmp v3 target-parameters *target-parameter-name*]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name]
notify-filter profile-name;
parameters {
    message-processing-model (v1 | v2c | V3);
    security-level (authentication | none | privacy);
    security-model (usm | v1 | v2c);
    security-name security-name;
}
```

This topic includes the following sections:

Applying the Trap Notification Filter

To apply the trap notification filter, include the **notify-filter** statement at the **[edit snmp v3 target-parameters *target-parameter-name*]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name]
notify-filter profile-name;
```

profile-name is the name of a configured notify filter. For information about configuring notify filters, see "Configuring the Trap Notification Filter" on page 132.

Configuring the Target Parameters

To configure target parameter properties, include the following statements at the **[edit snmp v3 target-parameters *target-parameter-name* parameters]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name parameters]
message-processing-model (v1 | v2c | v3);
security-level (authentication | none | privacy);
security-model (usm | v1 | v2c);
security-name security-name;
```

This section includes the following topics:

Configuring the Message Processing Model

The message processing model defines which version of SNMP to use when generating SNMP notifications. To configure the message processing model, include the **message-processing-model** statement at the **[edit snmp v3 target-parameters *target-parameter-name* parameters]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name parameters]
message-processing-model (v1 | v2c | v3);
```

- **v1**—SNMPv1 message processing model
- **v2c**—SNMPv2c message processing model
- **v3**—SNMPV3 message processing model

Configuring the Security Model

To define the security model to use when generating SNMP notifications, include the **security-model** statement at the **[edit snmp v3 target-parameters *target-parameter-name* parameters]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name parameters]
security-model (usm | v1 | v2c);
```

- **usm**—SNMPv3 security model
- **v1**—SNMPv1 security model
- **v2c**—SNMPv2c security model

Configuring the Security Level

The **security-level** statement specifies whether the trap is authenticated and encrypted before it is sent.

To configure the security level to use when generating SNMP notifications, include the **security-level** statement at the **[edit snmp v3 target-parameters *target-parameter-name* parameters]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name parameters]
security-level (authentication | none | privacy);
```

- **authentication**—Provides authentication but no encryption.
- **none**—No security. Provides no authentication and no encryption.
- **privacy**—Provides authentication and encryption.

NOTE: If you are configuring the SNMPv1 or SNMPV2c security model, use **none** as your security level. If you are configuring the SNMPv3 (USM) security model, use the **authentication** or **privacy** security level.

Configuring the Security Name

To configure the security name to use when generating SNMP notifications, include the **security-name** statement at the **[edit snmp v3 target-parameters *target-parameter-name* parameters]** hierarchy level:

```
[edit snmp v3 target-parameters target-parameter-name parameters]
security-name security-name;
```

If the USM security model is used, the **security-name** identifies the user that is used when the notification is generated. If the v1 or v2c security models are used, **security-name** identifies the SNMP community used when the notification is generated.

NOTE: The access privileges for the group associated with a security name must allow this notification to be sent.

If you are using the v1 or v2 security models, the security name at the **[edit snmp v3 vacm security-to-group]** hierarchy level must match the security name at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the SNMPv3 Trap Notification on page 131
- Configuring the Trap Notification Filter on page 132
- Configuring the Trap Target Address on page 133
- Configuring SNMP Informs on page 139
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring SNMP Informs

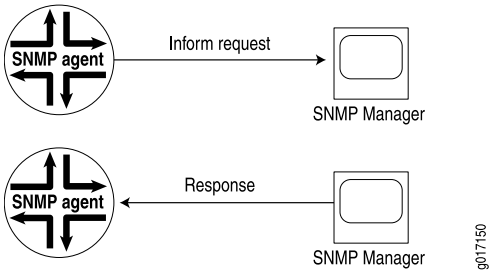
Junos OS supports two types of notifications: traps and informs. With traps, the receiver does not send any acknowledgment when it receives a trap. Therefore, the sender cannot determine if the trap was received. A trap may be lost because a problem occurred during transmission. To increase reliability, an inform is similar to a trap except that the inform is stored and retransmitted at regular intervals until one of these conditions occurs:

- The receiver (target) of the inform returns an acknowledgment to the SNMP agent.
- A specified number of unsuccessful retransmissions have been attempted and the agent discards the inform message.

If the sender never receives a response, the inform can be sent again. Thus, informs are more likely to reach their intended destination than traps are. Informs use the same communications channel as traps (same socket and port) but have different protocol data unit (PDU) types.

Informs are more reliable than traps, but they consume more network, router, and switch resources (see Figure 3 on page 140). Unlike a trap, an inform is held in memory until a response is received or the timeout is reached. Also, traps are sent only once, whereas an inform may be retried several times. Use informs when it is important that the SNMP manager receive all notifications. However, if you are more concerned about network traffic, or router and switch memory, use traps.

Figure 3: Inform Request and Response



For information about configuring SNMP traps, see "Configuring SNMPv3 Traps on a Device Running Junos OS" on page 129.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring the Remote Engine and Remote User on page 140
- Configuring the Inform Notification Type and Target Address on page 142
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the Remote Engine and Remote User

To send inform messages to an SNMPv3 user on a remote device, you must first specify the engine identifier for the SNMP agent on the remote device where the user resides. The remote engine ID is used to compute the security digest for authenticating and encrypting packets sent to a user on the remote host. When sending an inform message, the agent uses the credentials of the user configured on the remote engine (inform target).

To configure a remote engine and remote user to receive and respond to SNMP informs, include the following statements at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
usm {
  remote-engine engine-id {
    user username {
      authentication-md5 {
        authentication-key key;
      }
      authentication-none;
      authentication-sha {
        authentication-key key;
      }
      privacy-3des {
```

```
            privacy-key key;
          }
          privacy-aes128 {
            privacy-key key;
          }
          privacy-des {
            privacy-key key;
          }
          privacy-none;
        }
      }
    }
```

For informs, **remote-engine *engine-id*** is the identifier for the SNMP agent on the remote device where the user resides.

For informs, **user *username*** is the user on a remote SNMP engine who receives the informs.

Informs generated can be **unauthenticated**, **authenticated**, or **authenticated_and_encrypted**, depending on the security level of the SNMPv3 user configured on the remote engine (the inform receiver). The authentication key is used for generating message authentication code (MAC). The privacy key is used to encrypt the inform PDU part of the message.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring SNMP Informs on page 139
- Configuring the Inform Notification Type and Target Address on page 142
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: Configuring the Remote Engine ID and Remote Users on page 141

Example: Configuring the Remote Engine ID and Remote Users

The following example configures user **u10** located on remote engine **0x800007E5804089071BC6D10A41** and the user's authentication and privacy keys. The keys are autogenerated from the passwords entered by the command-line interface (CLI) user.

```
[edit snmp v3]
usm {
  remote-engine 800007E5804089071BC6D10A41 {
    user u10 {
      authentication-md5 {
        authentication-key "$9$D0jP536901Riktu1IcSwY2gUj5QF3
        /CYgQF/Cu0xN-bwgZGiqP5iH.5TF/9WLX7wYoaUkqfoaAp
        0BEhSreW87s24aUjsY4ZDjq.RhcyWLNdbg4Zs
        YJDHkTQ69Apu1EcyrvWQF/tuOREYg4ajHmPQF39
        Ygz3n6At8XxNYgik.PTz7-ikmfn6vW8XVw";
      }
    }
    privacy-des {
```

```
            privacy-key "$9$MZZXxdwYgJUjlKJGiH5T69Au0IrlM7NbeK24
            aJDjO1IRylM8Xbwg1R24aJDjHqm5n/Ap0ORhn6evLXbwmf5T
            /CRhSyKM5QEcleW87-Vbs4JGD.mT-VwgaZkqfTznAphSrlM8yr
            Wx7dsYTzF36AtuO1EcpuNdwYoa69CuRhcyleM8rlaZGjq.O1IEhr";
        }
    }
}
```

Related Documentation

- Configuring the Remote Engine and Remote User on page 140
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the Inform Notification Type and Target Address

To configure the inform notification type and target information, include the following statements at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
notify name {
    tag tag-name;
    type (trap | inform);
}
target-address target-address-name {
    address address;
    address-mask address-mask;
    logical-system (SNMP logical-system;
    port port-number;
    retry-count number;
    routing-instance instance;
    tag-list tag-list;
    target-parameters target-parameters-name;
    timeout seconds;
}
target-parameters target-parameters-name {
    notify-filter profile-name;
    parameters {
        message-processing-model (v1 | v2c | v3);
        security-level (authentication | none | privacy);
        security-model (usm | v1 | v2c);
        security-name security-name;
    }
}
```

notify *name* is the name assigned to the notification. Each notify entry name must be unique.

tag *tag-name* defines the target addresses that are sent this notification. The notification is sent to all target addresses that have this tag in their tag list. The ***tag-name*** is not included in the notification. For information about how to configure the tag list, see "Configuring the Trap Target Address" on page 134.

type inform is the type of notification.

target-address ***target-address-name*** identifies the target address. The target address defines a management application's address and parameters that are used to respond to informs.

timeout ***seconds*** is the number of seconds to wait for an acknowledgment. If no acknowledgment is received within the timeout period, the inform is retransmitted. The default timeout is **15** seconds.

retry-count ***number*** is the maximum number of times an inform is transmitted if no acknowledgment is received. The default is **3**. If no acknowledgment is received after the inform is transmitted the maximum number of times, the inform message is discarded.

message-processing-model defines which version of SNMP to use when SNMP notifications are generated. Informs require a **v3** message processing model.

security-model defines the security model to use when SNMP notifications are generated. Informs require a **usm** security model.

security-level specifies whether the inform is authenticated and encrypted before it is sent. For the **usm** security model, the security level must be one of the following:

- **authentication**—Provides authentication but no encryption.
- **privacy**—Provides authentication and encryption.

security-name identifies the username that is used when generating the inform.

Related Documentation

- Configuring SNMPv3 Traps on a Device Running Junos OS on page 129
- Configuring SNMP Informs on page 139
- Configuring the Remote Engine and Remote User on page 140
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: Configuring the Inform Notification Type and Target Address on page 143

Example: Configuring the Inform Notification Type and Target Address

In the following example, target **172.17.20.184** is configured to respond to informs. The inform timeout is **30** seconds and the maximum retransmit count is **3**. The inform is sent to all targets in the **tl1** list. The security model for the remote user is **usm** and the remote engine username is **u10**.

```
[edit snmp v3]
notify n1 {
    type inform;
    tag tl1;
}
notify-filter nf1 {
    oid .1.3 include;
}
target-address ta1 {
```

```
    address 172.17.20.184;
    retry-count 3;
    tag-list tl1;
    address-mask 255.255.255.0;
    target-parameters tp1;
    timeout 30;
}
target-parameters tp1 {
    parameters {
        message-processing-model v3;
        security-model usm;
        security-level privacy;
        security-name u10;
    }
    notify-filter nf1;
}
```

Related Documentation

- Configuring the Inform Notification Type and Target Address on page 142
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Configuring the SNMPv3 Community

The SNMP community defines the relationship between an SNMP server system and the client systems. This statement is optional.

To configure the SNMP community, include the **snmp-community** statement at the **[edit snmp v3]** hierarchy level:

```
[edit snmp v3]
snmp-community community-index;
```

community-index is the index for the SNMP community.

To configure the SNMP community properties, include the following statements at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level:

```
[edit snmp v3 snmp-community community-index]
community-name community-name;
context context-name;
security-name security-name;
tag tag-name;
```

This section includes the following topics:

- Configuring the Community Name on page 145
- Configuring the Context on page 145
- Configuring the Security Names on page 145
- Configuring the Tag on page 146

Configuring the Community Name

The community name defines the SNMP community. The SNMP community authorizes SNMPv1 or SNMPv2c clients. The access privileges associated with the configured security name define which MIB objects are available and the operations (read, write, or notify) allowed on those objects.

To configure the SNMP community name, include the **community-name** statement at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level:

```
[edit snmp v3 snmp-community community-index]
community-name community-name;
```

community-name is the community string for an SNMPv1 or SNMPv2c community.

If unconfigured, it is the same as the community index.

If the community name contains spaces, enclose it in quotation marks (“ “).

NOTE: Community names must be unique. You cannot configure the same community name at the **[edit snmp community]** and **[edit snmp v3 snmp-community *community-index*]** hierarchy levels. The configured community name at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level is encrypted. You cannot view the community name after you have configured it and committed your changes. In the command-line interface (CLI), the community name is concealed.

Configuring the Context

An SNMP context defines a collection of management information that is accessible to an SNMP entity. Typically, an SNMP entity has access to multiple contexts. A context can be a physical or logical system, a collection of multiple systems, or even a subset of a system. Each context in a management domain has a unique identifier.

To configure an SNMP context, include the **context *context-name*** statement at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level:

```
[edit snmp v3 snmp-community community-index]
context context-name;
```

NOTE: To query a routing instance or a logical system,

Configuring the Security Names

To assign a community string to a security name, include the **security-name** statement at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level:

```
[edit snmp v3 snmp-community community-index]
security-name security-name;
```

security-name is used when access control is set up. The **security-to-group** configuration at the **[edit snmp v3 vacm]** hierarchy level identifies the group.

NOTE: This security name must match the security name configured at the **[edit snmp v3 target-parameters *target-parameters-name* parameters]** hierarchy level when you configure traps.

Configuring the Tag

To configure the tag, include the **tag** statement at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level:

```
[edit snmp v3 snmp-community community-index]
tag tag-name;
```

tag-name identifies the address of managers that are allowed to use a community string.

Related Documentation

- Creating SNMPv3 Users on page 118
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116
- Example: SNMPv3 Community Configuration on page 146

Example: SNMPv3 Community Configuration

Define an SNMP community:

```
[edit snmp v3]
snmp-community index1 {
   community-name "$9$JOZi.QF/AtOz3"; # SECRET-DATA
   security-name john;
   tag router1; # Identifies managers that are allowed to use
   # a community string
   target-address ta1 {
      address 10.1.1.1;
      address-mask 255.255.255.0; # Defines the range of addresses
      port 162;
      tag-list router1;
      target-parameters tp1; # Applies configured target parameters
   }
}
```

Related Documentation

- Configuring the SNMPv3 Community on page 144
- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

Example: SNMPv3 Configuration

Define an SNMPv3 configuration:

```
[edit snmp]
engine-id {
   use-mac-address;
}
view jnxAlarms {
   oid 1.3.6.1.4.1.2636.3.4 include;
}
view interfaces {
   oid 1.3.6.1.2.1.2 include;
}
view ping-mib {
   oid 1.3.6.1.2.1.80 include;
}
[edit snmp v3]
notify n1 {
   tag router1; # Identifies a set of target addresses
   type trap;# Defines type of notification
}
notify n2 {
   tag host1;
   type trap;
}
notify-filter nf1 {
   oid .1 include; # Defines which traps to send
} # In this case, includes all traps
notify-filter nf2 {
   oid 1.3.6.1.4.1 include; # Sends enterprise-specific traps only
}
notify-filter nf3 {
   oid 1.3.6.1.2.1.1.5 include; # Sends BGP traps only
}
snmp-community index1 {
   community-name "$9$JOZi.QF/AtOz3"; # SECRET-DATA
   security-name john; # Matches the security name at the target parameters
   tag host1; # Finds the addresses that are allowed to be used with
}
target-address ta1 {# Associates the target address with the group
            # san-francisco.
   address 10.1.1.1;
   address-mask 255.255.255.0; # Defines the range of addresses
   port 162;
   tag-list router1;
   target-parameters tp1; # Applies configured target parameters
}
target-address ta2 {
   address 10.1.1.2;
   address-mask 255.255.255.0;
   port 162;
   tag-list host1;
   target-parameters tp2;
```

```
}
target-address ta3 {
  address 10.1.1.3;
  address-mask 255.255.255.0;
  port 162;
  tag-list “router1 host1”;
  target-parameters tp3;
}
target-parameters tp1 { # Defines the target parameters
  notify-filter nf1; # Specifies which notify filter to apply
  parameters {
    message-processing-model v1;
    security-model v1;
    security-level none;
    security-name john; # Matches the security name configured at the
  } # [edit snmp v3 snmp-community community-index hierarchy level.
}
target-parameters tp2 {
  notify-filter nf2;
  parameters {
    message-processing-model v1;
    security-model v1;
    security-level none;
    security-name john;
  }
}
target-parameters tp3 {
  notify-filter nf3;
  parameters {
    message-processing-model v1;
    security-model v1;
    security-level none;
    security-name john;
  }
}
usm {
  local-engine { #Defines authentication and encryption for SNMPv3 users
    user user1 {
      authentication-md5 {
        authentication-password authentication-password;
      }
      privacy-des {
        privacy-password privacy-password;
      }
    }
    user user2 {
      authentication-sha {
        authentication-password authentication-password;
      }
      privacy-none;
    }
    user user3 {
      authentication-none;
      privacy-none;
    }
    user user4 {
```

```
            authentication-sha {
              authentication-password authentication-password;
            }
            privacy-aes128 {
              privacy-password privacy-password;
            }
          }
          user user5 {
            authentication-sha {
              authentication-password authentication-password;
            }
            privacy-none;
          }
        }
      }
      vacm {
        access {
          group san-francisco { #Defines the access privileges for the group
            default-context-prefix { # called san-francisco
              security-model v1 {
                security-level none {
                  notify-view ping-mib;
                  read-view interfaces;
                  write-view jnxAlarms;
                }
              }
            }
          }
        }
        security-to-group {
          security-model v1 {
            security-name john { # Assigns john to the security group
              group san-francisco; # called san-francisco
            }
            security-name bob {
              group new-york;
            }
            security-name elizabeth {
              group chicago;
            }
          }
        }
      }
```

Related Documentation

- Complete SNMPv3 Configuration Statements on page 114
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

CHAPTER 8

SNMP Remote Operations and Support for Routing Instances

Identifying a Routing Instance

With this feature, routing instances are identified by either the context field in v3 requests or encoded in the community string in v1 or v2c requests.

When encoded in a community string, the routing instance name appears first and is separated from the actual community string by the @ character.

To avoid conflicts with valid community strings that contain the @ character, the community is parsed only if typical community string processing fails. For example, if a routing instance named **RI** is configured, an SNMP request with **RI@public** is processed within the context of the **RI** routing instance. Access control (views, source address restrictions, access privileges, and so on) is applied according to the actual community string (the set of data after the @ character—in this case **public**). However, if the community string **RI@public** is configured, the protocol data unit (PDU) is processed according to that community and the embedded routing instance name is ignored.

Logical systems perform a subset of the actions of a physical router and have their own unique routing tables, interfaces, policies, and routing instances. When a routing instance is defined within a logical system, the logical system name must be encoded along with the routing instance using a slash (/) to separate the two. For example, if the routing instance **RI** is configured within the logical system **LS**, that routing instance must be encoded within a community string as **LS/RI@public**. When a routing instance is configured outside a logical system (within the default logical system), no logical system name (or / character) is needed.

Also, when a logical system is created, a default routing instance (named **default**) is always created within the logical system. This name should be used when querying data

for that routing instance (for example, **LS/default@public**). For v3 requests, the name ***logical system/routing instance*** should be identified directly in the context field.

NOTE: To identify a virtual LAN (VLAN) spanning-tree instance (VSTP on MX Series 3D Universal Edge Routers), specify the routing instance name followed by a double colon (::) and the VLAN ID. For example, to identify VSTP instance for VLAN 10 in the global default routing instance, include **default::10@public** in the **context** (SNMPv3) or **community** (SNMPv1 or v2) string.

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Enabling SNMP Access over Routing Instances on page 152
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152

Enabling SNMP Access over Routing Instances

To enable SNMP managers in routing instances other than the default routing instance to access SNMP information, include the **routing-instance-access** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
routing-instance-access;
```

If this statement is not included in the SNMP configuration, SNMP managers from routing instances other than the default routing instance cannot access SNMP information.

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Identifying a Routing Instance on page 151
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152
- Configuring Access Lists for SNMP Access over Routing Instances on page 155

Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community

You can specify the routing instance along with the client information when you add a client to an SNMP community. To specify the routing instance to which a client belongs, include the **routing-instance** statement followed by the routing instance name and client information in the SNMP configuration.

The following example shows the configuration statement to add routing instance **test-ri** to SNMP community **community1**.

NOTE: Routing instances specified at the **[edit snmp community *community-name*]** hierarchy level are added to the default logical system in the community.

```
[edit snmp]
community community1 {
  clients {
    10.209.152.33/32;
  }
  routing-instance test-ri {
    clients {
      10.19.19.1/32;
    }
  }
}
```

If the routing instance is defined within a logical system, include the **routing-instance** statement at the [**edit snmp community *community-name* logical-system *logical-system-name***] hierarchy level, as in the following example:

```
[edit snmp]
community community1 {
  clients {
    10.209.152.33/32;
  }
  logical-system test-LS {
    routing-instance test-ri {
      clients {
        10.19.19.1/32;
      }
    }
  }
}
```

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Identifying a Routing Instance on page 151
- Enabling SNMP Access over Routing Instances on page 152
- Configuring Access Lists for SNMP Access over Routing Instances on page 155
- Example: Configuring Interface Settings for a Routing Instance on page 153

Example: Configuring Interface Settings for a Routing Instance

This example shows an **802.3ad ae0** interface configuration allocated to a routing instance named **INFrtd**:

```
[edit chassis]
aggregated-devices {
  ethernet {
    device-count 5;
  }
}
[edit interfaces ae0]
vlan-tagging;
aggregated-ether-options {
  minimum-links 2;
  link-speed 100m;
```

```
}
unit 0 {
   vlan-id 100;
   family inet {
      address 10.1.0.1/24;
   }
}
[edit interfaces fe-1/1/0]
fastether-options {
   802.3ad ae0;
}
[edit interfaces fe-1/1/1]
fastether-options {
   802.3ad ae0;
}
[edit routing-instances]
INFrtd {
   instance-type virtual-router;
   interface fe-1/1/0.0;
   interface fe-1/1/1.0;
   interface fe-1/1/5.0;
   interface ae0.0;
   protocols {
      ospf {
         area 0.0.0.0 {
            interface all;
         }
      }
   }
}
```

The following **snmpwalk** command shows how to retrieve SNMP-related information from **router1** and the 802.3ae bundle interface belonging to routing instance **INFrtd** with the SNMP community **public**:

```
router# snmpwalk -Os router1 INFrtd@public dot3adAggTable
dot3adAggMACAddress.59 = 0:90:69:92:93:f0
dot3adAggMACAddress.65 = 0:90:69:92:93:f0
dot3adAggActorSystemPriority.59 = 0
dot3adAggActorSystemPriority.65 = 0
dot3adAggActorSystemID.59 = 0:0:0:0:0:0
dot3adAggActorSystemID.65 = 0:0:0:0:0:0
dot3adAggAggregateOrIndividual.59 = true(1)
dot3adAggAggregateOrIndividual.65 = true(1)
dot3adAggActorAdminKey.59 = 0
dot3adAggActorAdminKey.65 = 0
dot3adAggActorOperKey.59 = 0
dot3adAggActorOperKey.65 = 0
dot3adAggPartnerSystemID.59 = 0:0:0:0:0:0
dot3adAggPartnerSystemID.65 = 0:0:0:0:0:0
dot3adAggPartnerSystemPriority.59 = 0
dot3adAggPartnerSystemPriority.65 = 0
dot3adAggPartnerOperKey.59 = 0
dot3adAggPartnerOperKey.65 = 0
dot3adAggCollectorMaxDelay.59 = 0
dot3adAggCollectorMaxDelay.65 = 0
```

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152

Configuring Access Lists for SNMP Access over Routing Instances

You can create and maintain access lists to manage access to SNMP information. Access list configuration enables you to allow or deny SNMP access to clients of a specific routing instance.

The following example shows how to create an access list:

```
[edit snmp]
routing-instance-access {
   access-list {
      ri1 restrict;
      ls1/default;
      ls1/ri2;
      ls1*;
   }
}
```

The configuration given in the example:

- Restricts clients in **ri1** from accessing SNMP information.
- Allows clients in **ls1/default**, **ls1/ri2**, and all other routing instances with names starting with **ls1** to access SNMP information.

You can use the wildcard character (*) to represent a string in the routing instance name.

NOTE: **You cannot restrict the SNMP manager of the default routing instance from accessing SNMP information.**

Related Documentation

- Understanding SNMP Support for Routing Instances on page 16
- Enabling SNMP Access over Routing Instances on page 152
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152

CHAPTER 9

Remote Monitoring and Health Monitoring

Understanding RMON Alarms and Events Configuration

Junos OS supports monitoring routers from remote devices. These values are measured against thresholds and trigger events when the thresholds are crossed. You configure remote monitoring (RMON) alarm and event entries to monitor the value of a MIB object.

To configure RMON alarm and event entries, you include statements at the **[edit snmp]** hierarchy level of the configuration:

```
[edit snmp]
rmon {
  alarm index {
    description text-description;
    falling-event-index index;
    falling-threshold integer;
    falling-threshold-interval seconds;
    interval seconds;
    rising-event-index index;
    rising-threshold integer;
    request-type (get-next-request | get-request | walk-request);
    sample-type (absolute-value | delta-value);
    startup-alarm (falling-alarm | rising-alarm | rising-or-falling-alarm);
    syslog-subtag syslog-subtag;
    variable oid-variable;
    event index {
      community community-name;
      description description;
      type type;
    }
  }
}
```

Related Documentation

- Understanding RMON Alarms on page 65
- Understanding RMON Events on page 76
- Configuring an Alarm Entry and Its Attributes on page 158
- Configuring an Event Entry and Its Attributes on page 162
- Using alarmTable to Monitor MIB Objects on page 318
- Using eventTable to Log Alarms on page 322
- Minimum RMON Alarm and Event Entry Configuration on page 324

Configuring an Alarm Entry and Its Attributes

An alarm entry monitors the value of a MIB variable. You can configure how often the value is sampled, the type of sampling to perform, and what event to trigger if a threshold is crossed.

This section discusses the following topics:

Configuring the Alarm Entry

An alarm entry monitors the value of a MIB variable. The **rising-event-index**, **rising-threshold**, **sample-type**, and **variable** statements are mandatory. All other statements are optional.

To configure the alarm entry, include the **alarm** statement and specify an index at the **[edit snmp rmon]** hierarchy level:

```
[edit snmp rmon]
alarm index {
    description description;
    falling-event-index index;
    falling-threshold integer;
    falling-threshold-interval seconds;
    interval seconds;
    rising-event-index index;
```

```
    rising-threshold integer;
    sample-type (absolute-value | delta-value);
    startup-alarm (falling-alarm | rising alarm | rising-or-falling-alarm);
    variable oid-variable;
}
```

index is an integer that identifies an alarm or event entry.

Configuring the Description

The description is a text string that identifies the alarm entry.

To configure the description, include the **description** statement and a description of the alarm entry at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
description description;
```

Configuring the Falling Event Index or Rising Event Index

The falling event index identifies the event entry that is triggered when a falling threshold is crossed. The rising event index identifies the event entry that is triggered when a rising threshold is crossed.

To configure the falling event index or rising event index, include the **falling-event-index** or **rising-event-index** statement and specify an index at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
falling-event-index index;
rising-event-index index;
```

index can be from 0 through 65,535. The default for both the falling and rising event index is 0.

Configuring the Falling Threshold or Rising Threshold

The falling threshold is the lower threshold for the monitored variable. When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval is greater than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is less than or equal to this threshold, and the associated startup alarm is equal to **falling-alarm** or **rising-or-falling-alarm**. After a falling event is generated, another falling event cannot be generated until the sampled value rises above this threshold and reaches the rising threshold. You must specify the falling threshold as an integer. Its default is 20 percent less than the rising threshold.

By default, the rising threshold is 0. The rising threshold is the upper threshold for the monitored variable. When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval is less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is greater than or equal to this threshold, and the associated **startup-alarm** is equal to **rising-alarm** or **rising-or-falling-alarm**. After a rising event is generated, another rising event cannot be generated until the sampled value falls below this threshold and reaches the falling threshold. You must specify the rising threshold as an integer.

To configure the falling threshold or rising threshold, include the **falling-threshold** or **rising-threshold** statement at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
falling-threshold integer;
rising-threshold integer;
```

integer can be a value from -2,147,483,647 through 2,147,483,647.

Configuring the Interval

The interval represents the period of time, in seconds, over which the monitored variable is sampled and compared with the rising and falling thresholds.

To configure the interval, include the **interval** statement and specify the number of seconds at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
interval seconds;
```

seconds can be a value from 1 through 2,147,483,647. The default is 60 seconds.

Configuring the Falling Threshold Interval

The falling threshold interval represents the interval between samples when the rising threshold is crossed. Once the alarm crosses the falling threshold, the regular sampling interval is used.

NOTE: **You cannot configure the falling threshold interval for alarms that have the request type set to walk-request.**

To configure the falling threshold interval, include the **falling-threshold interval** statement at the **[edit snmp rmon alarm *index*]** hierarchy level and specify the number of seconds:

```
[edit snmp rmon alarm index]
falling-threshold-interval seconds;
```

seconds can be a value from 1 through 2,147,483,647. The default is 60 seconds.

Configuring the Request Type

By default an RMON alarm can monitor only one object instance (as specified in the configuration). You can configure a **request-type** statement to extend the scope of the RMON alarm to include all object instances belonging to a MIB branch or to include the next object instance after the instance specified in the configuration.

To configure the request type, include the **request-type** statement at the **[edit snmp rmon alarm *index*]** hierarchy level and specify **get-next-request**, **get-request**, or **walk-request**:

```
[edit snmp rmon alarm index]
request-type (get-next-request | get-request | walk-request);
```

walk extends the RMON alarm configuration to all object instances belonging to a MIB branch. **next** extends the RMON alarm configuration to include the next object instance after the instance specified in the configuration.

Configuring the Sample Type

The sample type identifies the method of sampling the selected variable and calculating the value to be compared against the thresholds. If the value of this object is **absolute-value**, the value of the selected variable is compared directly with the thresholds at the end of the sampling interval. If the value of this object is **delta-value**, the value of the selected variable at the last sample is subtracted from the current value, and the difference is compared with the thresholds.

To configure the sample type, include the **sample-type** statement and specify the type of sample at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
sample-type (absolute-value | delta-value);
```

- **absolute-value**—Actual value of the selected variable is compared against the thresholds.
- **delta-value**—Difference between samples of the selected variable is compared against the thresholds.

Configuring the Startup Alarm

The startup alarm identifies the type of alarm that can be sent when this entry is first activated. You can specify it as **falling-alarm**, **rising-alarm**, or **rising-or-falling-alarm**.

To configure the startup alarm, include the **startup-alarm** statement and specify the type of alarm at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
startup-alarm (falling-alarm | rising-alarm | rising-or-falling-alarm);
```

- **falling-alarm**—Generated if the first sample after the alarm entry becomes active is less than or equal to the falling threshold.
- **rising-alarm**—Generated if the first sample after the alarm entry becomes active is greater than or equal to the rising threshold.
- **rising-or-falling-alarm**—Generated if the first sample after the alarm entry becomes active satisfies either of the corresponding thresholds.

The default is **rising-or-falling-alarm**.

Configuring the System Log Tag

The **syslog-subtag** statement specifies the tag to be added to the system log message. You can specify a string of not more than 80 uppercase characters as the system log tag.

To configure the system log tag, include the **syslog-subtag** statement at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
syslog-subtag syslog-subtag;
```

Configuring the Variable

The variable identifies the MIB object that is being monitored.

To configure the variable, include the **variable** statement and specify the object identifier or object name at the **[edit snmp rmon alarm *index*]** hierarchy level:

```
[edit snmp rmon alarm index]
variable oid-variable;
```

oid-variable is a dotted decimal (for example, **1.3.6.1.2.1.2.1.2.2.1.10.1**) or MIB object name (for example, **ifInOctets.1**).

Configuring an Event Entry and Its Attributes

An event entry generates a notification for an alarm entry when its rising or falling threshold is crossed. You can configure the type of notification that is generated. To configure the event entry, include the **event** statement at the **[edit snmp rmon]** hierarchy level. All statements except the **event** statement are optional.

```
[edit snmp rmon]
event index {
  community community-name;
  description description;
  type type;
}
```

index identifies an entry event.

community-name is the trap group that is used when generating a trap. If that trap group has the **rmon-alarm** trap category configured, a trap is sent to all the targets configured for that trap group. The community string in the trap matches the name of the trap group. If nothing is configured, all the trap groups are examined, and traps are sent using each group with the **rmon-alarm** category set.

description is a text string that identifies the entry.

The ***type*** variable of an event entry specifies where the event is to be logged. You can specify the type as one of the following:

- **log**—Adds the event entry to the **logTable**.
- **log-and-trap**—Sends an SNMP trap and creates a log entry.
- **none**—Sends no notification.
- **snmptrap**—Sends an SNMP trap.

The default for the event entry type is **log-and-trap**.

Related Documentation

- Understanding RMON Alarms and Events Configuration on page 157
- Understanding RMON Alarms on page 65
- Understanding RMON Events on page 76
- Configuring an Alarm Entry and Its Attributes on page 158

- Minimum RMON Alarm and Event Entry Configuration on page 324
- Example: Configuring an RMON Alarm and Event Entry on page 163

Example: Configuring an RMON Alarm and Event Entry

Configure an RMON alarm and event entry:

```
[edit snmp]
rmon {
   alarm 100 {
      description “input traffic on fxp0”;
      falling-event-index 100;
      falling-threshold 10000;
      interval 60;
      rising-event-index 100;
      rising-threshold 100000;
      sample-type delta-value;
      startup-alarm rising-or-falling-alarm;
      variable ifInOctets.1;
   }
   event 100 {
      community bedrock;
      description” emergency events”;
      type log-and-trap;
   }
}
```

Related Documentation

- Understanding RMON Alarms and Events Configuration on page 157
- Configuring an Alarm Entry and Its Attributes on page 158
- Configuring an Event Entry and Its Attributes on page 162

Configuring Health Monitoring on Devices Running Junos OS

As the number of devices managed by a typical network management system (NMS) grows and the complexity of the devices themselves increases, it becomes increasingly impractical for the NMS to use polling to monitor the devices. A more scalable approach is to rely on network devices to notify the NMS when something requires attention.

On Juniper Networks routers, RMON alarms and events provide much of the infrastructure needed to reduce the polling overhead from the NMS. However, with this approach, you must set up the NMS to configure specific MIB objects into RMON alarms. This often requires device-specific expertise and customizing of the monitoring application. In addition, some MIB object instances that need monitoring are set only at initialization or change at runtime and cannot be configured in advance.

To address these issues, the health monitor extends the RMON alarm infrastructure to provide predefined monitoring for a selected set of object instances (for file system usage, CPU usage, and memory usage) and includes support for unknown or dynamic object instances (such as Junos OS processes).

Health monitoring is designed to minimize user configuration requirements. To configure health monitoring entries, include the **health-monitor** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
health-monitor {
  falling-threshold percentage;
  interval seconds;
  rising-threshold percentage;
}
```

You can use the **show snmp health-monitor** operational command to view information about health monitor alarms and logs.

This topic describes the minimum required configuration and discusses the following tasks for configuring the health monitor:

Monitored Objects

When you configure the health monitor, monitoring information for certain object instances is available, as shown in Table 12 on page 164.

Table 12: Monitored Object Instances

Object	Description
jnxHrStoragePercentUsed.1	Monitors the following file system on the router or switch: **/dev/ad0s1a:** This is the root file system mounted on */*.
jnxHrStoragePercentUsed.2	Monitors the following file system on the router or switch: **/dev/ad0s1e:** This is the configuration file system mounted on **/config**.
jnxOperatingCPU (RE0) **jnxOperatingCPU (RE1)**	Monitors CPU usage for Routing Engines (**RE0** and **RE1**). The index values assigned to Routing Engines depend on whether the Chassis MIB uses a zero-based or ones-based indexing scheme. Because the indexing scheme is configurable, the proper index is determined when the router or switch is initialized and when there is a configuration change. If the router or switch has only one Routing Engine, the alarm entry monitoring **RE1** is removed after five failed attempts to obtain the CPU value.

Table 12: Monitored Object Instances *(continued)*

Object	Description
jnxOperatingBuffer (RE0)	Monitors the amount of memory available on Routing Engines (**RE0** and **RE1**). Because the indexing of this object is identical to that used for **jnxOperatingCPU**, index values are adjusted depending on the indexing scheme used in the Chassis MIB. As with **jnxOperatingCPU**, the alarm entry monitoring **RE1** is removed if the router or switch has only one Routing Engine.
jnxOperatingBuffer (RE1)	
sysApplElmtRunCPU	Monitors the CPU usage for each Junos OS process (also called daemon). Multiple instances of the same process are monitored and indexed separately.
sysApplElmtRunMemory	Monitors the memory usage for each Junos OS process. Multiple instances of the same process are monitored and indexed separately.

Minimum Health Monitoring Configuration

To enable health monitoring on the router or switch, include the **health-monitor** statement at the **[edit snmp]** hierarchy level:

```
[edit snmp]
health-monitor;
```

Configuring the Falling Threshold or Rising Threshold

The falling threshold is the lower threshold (expressed as a percentage of the maximum possible value) for the monitored variable. When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval is greater than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is less than or equal to this threshold. After a falling event is generated, another falling event cannot be generated until the sampled value rises above this threshold and reaches the rising threshold. You must specify the falling threshold as a percentage of the maximum possible value. The default is **70** percent.

By default, the rising threshold is **80** percent of the maximum possible value for the monitored object instance. The rising threshold is the upper threshold for the monitored variable. When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval is less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is greater than or equal to this threshold. After a rising event is generated, another rising event cannot be generated until the sampled value falls below this threshold and reaches the falling threshold. You must specify the rising threshold as a percentage of the maximum possible value for the monitored variable.

To configure the falling threshold or rising threshold, include the **falling-threshold** or **rising-threshold** statement at the **[edit snmp health-monitor]** hierarchy level:

```
[edit snmp health-monitor]
falling-threshold percentage;
rising-threshold percentage;
```

percentage can be a value from **1** through **100**.

The falling and rising thresholds apply to all object instances monitored by the health monitor.

Configuring the Interval

The interval represents the period of time, in seconds, over which the object instance is sampled and compared with the rising and falling thresholds.

To configure the interval, include the **interval** statement and specify the number of seconds at the **[edit snmp health-monitor]** hierarchy level:

```
[edit snmp health-monitor]
interval seconds;
```

seconds can be a value from **1** through **2147483647**. The default is **300** seconds (5 minutes).

Log Entries and Traps

The system log entries generated for any health monitor events (thresholds crossed, errors, and so on) have a corresponding **HEALTHMONITOR** tag rather than a generic **SNMPD_RMON_EVENTLOG** tag. However, the health monitor sends generic RMON **risingThreshold** and **fallingThreshold** traps.

Related Documentation

- Understanding RMON Alarms and Events Configuration on page 157
- Configuring an Alarm Entry and Its Attributes on page 158
- Configuring an Event Entry and Its Attributes on page 162
- Example: Configuring Health Monitoring on page 166
- Understanding Device Management Functions in Junos OS on page 3

Example: Configuring Health Monitoring

Configure the health monitor:

```
[edit snmp]
health-monitor {
   falling-threshold 85;
   interval 600;
   rising-threshold 75;
}
```

In this example, the sampling interval is every **600** seconds (10 minutes), the falling threshold is **85** percent of the maximum possible value for each object instance monitored, and the rising threshold is **75** percent of the maximum possible value for each object instance monitored.

Related Documentation

- Configuring Health Monitoring on Devices Running Junos OS on page 163

CHAPTER 10

Accounting, Source Class Usage, and Destination Class Usage Options

Configuration Statements at the [edit accounting-options] Hierarchy Level

This topic shows all possible configuration statements at the **[edit accounting-options]** hierarchy level and their level in the configuration hierarchy. When you are configuring Junos OS, your current hierarchy level is shown in the banner on the line preceding the **user@host#** prompt.

```
[edit]
accounting-options {
   class-usage-profile profile-name {
      file filename;
      interval minutes;
      destination-classes {
         destination-class-name;
      }
      source-classes {
         source-class-name;
      }
   }
   file filename {
```

```
        archive-sites {
        }
        files number;
        nonpersistent;
        size bytes;
        start-time time;
        transfer-interval minutes;
    }
    filter-profile profile-name {
        counters {
            counter-name;
        }
        file filename;
        interval minutes;
    }
}
interface-profile profile-name {
    fields {
        field-name;
    }
    file filename;
    interval minutes;
}
mib-profile profile-name {
    file filename;
    interval seconds;
    object-names {
        mib-object-name;
    }
    operation operation-name;
}
routing-engine-profile profile-name {
    fields {
        field-name;
    }
    file filename;
    interval minutes;
}
```

Related Documentation

- Accounting Options Overview on page 81
- Accounting Options Configuration on page 168

Accounting Options Configuration

This topic contains the following sections:

Accounting Options—Full Configuration

To configure accounting options, include the following statements at the **[edit accounting-options]** hierarchy level:

```
accounting-options {
  class-usage-profile profile-name {
    file filename;
    interval minutes;
    destination-classes {
      destination-class-name;
    }
    source-classes {
      source-class-name;
    }
    file filename {
      archive-sites {
        site-name;
      }
      files number;
      nonpersistent;
      size bytes;
      source-classes time
      transfer-interval minutes;
    }
    filter-profile profile-name {
      counters {
        counter-name;
      }
      file filename;
      interval minutes;
    }
  }
  interface-profile profile-name {
    fields {
      field-name;
    }
    file filename;
    interval minutes;
  }
  mib-profile profile-name {
    file filename;
    interval (Accounting Options) seconds;
    object-names {
      mib-object-name;
    }
    operation operation-name;
  }
  routing-engine-profile profile-name {
    fields {
      field-name;
    }
    file filename;
    interval minutes;
  }
```

```
}
```

By default, accounting options are disabled.

NOTE: Do not configure MIB objects related to interface octets or packets for a MIB profile, because it can cause the SNMP walk or a CLI show command to time out.

Minimum Accounting Options Configuration

To enable accounting options on the router, you must perform at least the following tasks:

- Configure accounting options by including a **file** statement and one or more **source-class-usage**, **destination-class-profile**, **filter-profile**, **interface-profile**, **mib-profile**, or **routing-engine-profile** statements at the **[edit accounting-options]** hierarchy level:

```
[edit]
accounting-options {
  class-usage-profile profile-name {
    file filename;
    interval minutes;
    source-classes {
      source-class-name;
      destination-classes {
        destination-class-name;
      }
    }
    file filename {
      archive-sites {
        site-name;
      }
      files number;
      size bytes;
      transfer-interval minutes;
    }
    filter-profile profile-name {
      counters {
        counter-name;
      }
      file filename;
      interval minutes;
    }
    interface-profile profile-name {
      fields {
        field-name;
      }
      file filename;
      interval (Accounting Options) minutes;
    }
    mib-profile profile-name {
      file filename;
      interval minutes;
      object-names {
```

```
        mib-object-name;
      }
      operation operation-name;
    }
    routing-engine-profile profile-name {
      fields {
        field-name;
      }
      file filename;
      interval minutes;
    }
  }
}
```

- Apply the profiles to the chosen interfaces or filters.

  Apply an interface profile to a physical or logical interface by including the **accounting-profile** statement at either the **[edit interfaces *interface-name*]** or the **[edit interfaces *interface-name* unit *logical-unit-number*]** hierarchy level. For more information about interface profiles, see the *Junos OS Network Interfaces Library for Routing Devices*.

  ```
  [edit interfaces]
  interface-name {
    accounting-profile profile-name;
    unit logical-unit-number {
      accounting-profile profile-name;
    }
  }
  ```

  NOTE: You do not apply destination class profiles to interfaces. Although the interface needs to have the **destination-class-usage** statement configured, the destination class profile automatically finds all interfaces with the destination class configured.

  Apply a filter profile to a firewall filter by including the **accounting-profile** statement at the **[edit firewall filter *filter-name*]** hierarchy level:

  ```
  [edit firewall]
  filter filter-name {
    accounting-profile profile-name;
  }
  ```

  You do not need to apply the Routing Engine profile to an interface because the statistics are collected on the Routing Engine itself.

Related Documentation

- Accounting Options Overview on page 81
- Understanding Device Management Functions in Junos OS on page 3
- Configuring Accounting-Data Log Files on page 172
- Configuring the Interface Profile on page 175
- Configuring the Filter Profile on page 177
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167

Configuring Accounting-Data Log Files

An accounting profile specifies what statistics should be collected and written to a log file. To configure an accounting-data log file, include the **file** statement at the **[edit accounting-options]** hierarchy level:

```
[edit accounting-options]
file filename {
    archive-sites {
        site-name;
    }
    files number;
    nonpersistent;
    size bytes;
    start-time time;
    transfer-interval minutes;
}
```

filename is the name of the file in which to write accounting data.

If the filename contains spaces, enclose it in quotation marks (" "). The filename cannot contain a forward slash (;/;). The file is created in the **/var/log** directory and can contain data from multiple profiles.

All accounting-data log files include header and trailer sections that start with a **#** in the first column. The header contains the file creation time, the hostname, and the columns that appear in the file. The trailer contains the time that the file was closed.

Whenever any configured value changes that affects the columns in a file, the file creates a new profile layout record that contains a new list of columns.

You must configure the file size; all other properties are optional.

- Configuring the Storage Location of the File on page 172
- Configuring the Maximum Size of the File on page 173
- Configuring the Maximum Number of Files on page 173
- Configuring the Start Time for File Transfer on page 173
- Configuring the Transfer Interval of the File on page 173
- Configuring Archive Sites on page 174

Configuring the Storage Location of the File

On J Series Services Routers, the files are stored by default on the compact flash drive. To configure the storage location of the files in the **mfs/var/log** directory (on DRAM) instead of the **cf/var/log** directory (on the compact flash drive), include the **nonpersistent** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
nonpersistent;
```

This feature is useful for minimizing read/write traffic on the router's compact flash drive.

NOTE: If log files for accounting data are stored on DRAM, these files are lost when you reboot the router. Therefore, you should back up these files periodically.

Configuring the Maximum Size of the File

To configure the maximum size of the files, include the **size** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
size bytes;
```

The **size** statement is the maximum size of the log file, in bytes, kilobytes (KB), megabytes (MB), or gigabytes (GB). The minimum value for ***bytes*** is 256 KB. You must configure ***bytes***; the remaining attributes are optional.

Configuring the Maximum Number of Files

To configure the maximum number of files, include the **files** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
files number;
```

When a log file (for example, **profilelog**) reaches its maximum size, it is renamed **profilelog.0**, then **profilelog.1**, and so on, until the maximum number of log files is reached. Then the oldest log file is overwritten. The minimum value for ***number*** is 3 and the default value is 10.

Configuring the Start Time for File Transfer

To configure the start time for transferring files, include the **start-time** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
start-time time;
```

The start-time statement specifies a start time for file transfer (***YYYY-MM-DD.hh:mm***). For example, 10:00 a.m. on January 30, 2007 is represented as **2007-01-30.10:00**.

Configuring the Transfer Interval of the File

To configure the transfer interval of the files, include the **transfer-interval** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
transfer-interval minutes;
```

The range for **transfer-interval** is 5 through 2880 minutes. The default is 30 minutes.

TIP:

Junos OS saves the existing log file and creates a new file at the configured transfer-intervals irrespective of:

- Whether the file has reached the maximum size or not
- Whether an archive site is configured or not

When you have a relatively smaller transfer-interval configured and if no archive site is configured, there is a possibility of losing data as Junos OS overwrites the log files when the maximum number of log files is reached. To ensure that the log information is saved for a reasonably long time:

- Configure an archive site to archive the log files every time a new log file is created.
- Configure the maximum value (2880 minutes) for **transfer-interval** so that new files are created less frequently; that is, only when the file exceeds the maximum size limit or once in 2 days.

Configuring Archive Sites

After a file reaches its maximum size or the **transfer-interval** time is exceeded, the file is closed, renamed, and, if you configured an archive site, transferred to a remote host. To configure archive sites, include the **archive-sites** statement at the **[edit accounting-options file *filename*]** hierarchy level:

```
[edit accounting-options file filename]
archive-sites {
  site-name;
}
```

site-name is any valid FTP URL. For more information about specifying valid FTP URLs, see the *Junos OS Administration Library for Routing Devices*. You can specify more than one URL, in any order. When a file is archived, the router or switch attempts to transfer the file to the first URL in the list, trying the next site in the list only if the transfer does not succeed. The log file is stored at the archive site with a filename of the format ***router-name_log-filename_timestamp***.

Related Documentation

- Accounting Options Overview on page 81
- Understanding Device Management Functions in Junos OS on page 3
- Accounting Options Configuration on page 168
- Configuring the Interface Profile on page 175
- Configuring the Filter Profile on page 177
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167

Configuring the Interface Profile

An interface profile specifies the information collected and written to a log file. You can configure a profile to collect error and statistic information for input and output packets on a particular physical or logical interface.

To configure an interface profile, include the **interface-profile** statement at the **[edit accounting-options]** hierarchy level:

```
[edit accounting-options]
interface-profile profile-name {
  fields {
    field-name;
  }
  file filename;
  interval minutes;
}
```

By default, the Packet Forwarding Engine (PFE) periodically collects the statistics for all interfaces. To improve the performance, you can optionally disable the periodic refresh by including the **periodic-refresh disable** statement at the **[edit accounting-options]** hierarchy level.

Each accounting profile must have a unique ***profile-name***. To apply a profile to a physical or logical interface, include the **accounting-profile** statement at either the **[edit interfaces *interface-name*]** or the **[edit interfaces *interface-name* unit *logical-unit-number*]** hierarchy level. You can also apply an accounting profile at the **[edit firewall family *family-type* filter *filter-name*]** hierarchy level. For more information, see the *Routing Policy Feature Guide for Routing Devices*.

To configure an interface profile, perform the tasks described in the following sections:

- Configuring Fields on page 175
- Configuring the File Information on page 175
- Configuring the Interval on page 176
- Example: Configuring the Interface Profile on page 176

Configuring Fields

An interface profile must specify what statistics are collected. To configure which statistics should be collected for an interface, include the **fields** statement at the **[edit accounting-options interface-profile *profile-name*]** hierarchy level:

```
[edit accounting-options interface-profile profile-name]
fields {
  field-name;
}
```

Configuring the File Information

Each accounting profile logs its statistics to a file in the **/var/log** directory.

To configure which file to use, include the **file** statement at the **[edit accounting-options interface-profile *profile-name*]** hierarchy level:

```
[edit accounting-options interface-profile profile-name]
file filename;
```

You must specify a **file** statement for the interface profile that has already been configured at the **[edit accounting-options]** hierarchy level.

Configuring the Interval

Each interface with an accounting profile enabled has statistics collected once per interval time specified for the accounting profile. Statistics collection time is scheduled evenly over the configured interval. To configure the interval, include the **interval** statement at the **[edit accounting-options interface-profile *profile-name*]** hierarchy level:

```
[edit accounting-options interface-profile profile-name]
interval minutes;
```

NOTE: The minimum interval allowed is 1 minute. Configuring a low interval in an accounting profile for a large number of interfaces might cause serious performance degradation.

The range for the **interval** statement is 1 through 2880 minutes. The default is 30 minutes.

Example: Configuring the Interface Profile

Configure the interface profile:

```
[edit]
accounting-options {
    file if_stats {
        size 40 files 5;
    }
    interface-profile if_profile1 {
        file if_stats;
        interval 30;
        fields {
            input-bytes;
            output-bytes;
            input-packets;
            output-packets;
            input-multicast;
            output-multicast;
        }
    }
    interface-profile if_profile2 {
        file if_stats;
        interval 30;
        fields {
            input-bytes;
            output-bytes;
            input-packets;
            output-packets;
            input-multicast;
```

```
        output-multicast;
      }
    }
    interfaces {
      xe-1/0/0 {
        accounting-profile if_profile1;
        unit 0 {
          accounting-profile if_profile2;
          ...
        }
      }
    }
  }
```

The two interface profiles, **if-profile1** and **if-profile2**, write data to the same file, **if-stats**. The **if-stats** file might look like the following:

```
#FILE CREATED 976823478 2000-12-14-19:51:18
#hostname host
#profile-layout
if_profile2,epoch-timestamp,interface-name,snmp-index,input-bytes,output-bytes,
input-packets,output-packets,input-multicast,output-multicast
#profile-layout
if_profile1,epoch-timestamp,interface-name,snmp-index,input-bytes,output-bytes,
input-packets
if_profile2,976823538,xe-1/0/0.0,8,134696815,3681534,501088,40723,0,0
if_profile1,976823538,xe-1/0/0,7,134696815,3681534,501088
...
#FILE CLOSED 976824378 2000-12-14-20:06:18
```

Related Documentation

- Accounting Options Overview on page 81
- Understanding Device Management Functions in Junos OS on page 3
- Accounting Options Configuration on page 168
- Configuring Accounting-Data Log Files on page 172
- Configuring the Filter Profile on page 177
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167

Configuring the Filter Profile

A filter profile specifies error and statistics information collected and written to a file. A filter profile must specify counter names for which statistics are collected.

To configure a filter profile, include the **filter-profile** statement at the **[edit accounting-options]** hierarchy level:

```
[edit accounting-options]
filter-profile profile-name {
  counters {
    counter-name;
  }
  file filename;
  interval minutes;
```

```
}
```

To apply the filter profile, include the **accounting-profile** statement at the **[edit firewall filter *filter-name*]** hierarchy level. For more information about firewall filters, see the *Junos OS Network Interfaces Library for Routing Devices*.

To configure a filter profile, perform the tasks described in the following sections:

- Configuring the Counters on page 178
- Configuring the File Information on page 178
- Configuring the Interval on page 179

Configuring the Counters

Statistics are collected for all counters specified in the filter profile. To configure the counters, include the **counters** statement at the **[edit accounting-options filter-profile *profile-name*]** hierarchy level:

```
[edit accounting-options filter-profile profile-name]
counters {
}
```

Configuring the File Information

Each accounting profile logs its statistics to a file in the **/var/log** directory.

To configure which file to use, include the **file** statement at the **[edit accounting-options filter-profile *profile-name*]** hierarchy level:

```
[edit accounting-options filter-profile profile-name]
file filename;
```

You must specify a filename for the filter profile that has already been configured at the **[edit accounting-options]** hierarchy level.

NOTE: The limit on the total number of characters per line in a log file equals 1023. If this limit is exceeded, the output written to the log file is incomplete. Ensure that you limit the number of counters or requested data so that this character limit is not exceeded.

NOTE: If the configured file size or transfer interval is exceeded, Junos OS closes the file and starts a new one. By default, the transfer interval value is 30 minutes. If the transfer interval is not configured, Junos OS closes the file and starts a new one when the file size exceeds its configured value or the default transfer interval value exceeds 30 minutes. To avoid transferring files every 30 minutes, specify a different value for the transfer interval.

Configuring the Interval

Each filter with an accounting profile enabled has statistics collected once per interval time specified for the accounting profile. Statistics collection time is scheduled evenly over the configured interval. To configure the interval, include the **interval** statement at the **[edit accounting-options filter-profile *profile-name*]** hierarchy level:

```
[edit accounting-options filter-profile profile-name]
interval;
```

NOTE: The minimum interval allowed is 1 minute. Configuring a low interval in an accounting profile for a large number of filters might cause serious performance degradation.

The range for the **interval** statement is 1 through 2880 minutes. The default is 30 minutes.

Related Documentation

- Accounting Options Overview on page 81
- Understanding Device Management Functions in Junos OS on page 3
- Accounting Options Configuration on page 168
- Configuring Accounting-Data Log Files on page 172
- Configuring the Interface Profile on page 175
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167
- Example: Configuring a Filter Profile on page 179
- Example: Configuring Interface-Specific Firewall Counters and Filter Profiles on page 180

Example: Configuring a Filter Profile

Configure a filter profile:

```
[edit]
accounting-options {
   file fw_accounting {
      size 500k files 4;
   }
   filter-profile fw_profile1 {
      file fw_accounting;
      interval 60;
      counters {
         counter1;
         counter2;
         counter3;
      }
   }
}
firewall {
   filter myfilter {
      accounting-profile fw_profile1;
```

```
    ...
    term accept-all {
      then {
        count counter1;
        accept;
      }
    }
  }
}
```

The filter profile, **fw-profile1**, writes data to the file **fw_accounting**. The file might look like the following:

```
#FILE CREATED 976825278 2000-12-14-20:21:18
#hostname host
#profile-layout
fw_profile1,epoch-timestamp,filter-name,counter-name,packet-count,byte-count
fw_profile1,976826058,myfilter,counter1,163,10764
...
#FILE CLOSED 976826178 2000-12-14-20:36:18
```

Related Documentation

- Configuring the Filter Profile on page 177
- Example: Configuring Interface-Specific Firewall Counters and Filter Profiles on page 180

Example: Configuring Interface-Specific Firewall Counters and Filter Profiles

To collect and log count statistics collected by firewall filters on a per-interface basis, you must configure a filter profile and include the interface-specific statement at the **[edit firewall filter *filter-name*]** hierarchy level.

Configure the firewall filter accounting profile:

```
[edit accounting-options]
file cust1_accounting {
  size 500k;
}
filter-profile cust1_profile {
  file cust1_accounting;
  interval 1;
  counters {
    r1;
  }
}
```

Configure the interface-specific firewall counter:

```
[edit firewall]
filter f3 {
  accounting-profile cust1_profile;
  interface-specific;
  term f3-term {
    then {
      count r1;
      accept;
    }
```

```
    }
  }
```

Apply the firewall filter to an interface:

```
[edit interfaces]
xe-1/0/0 {
  unit 0 {
    family inet {
      filter {
        input f3;
        output f3;
      }
      address 20.20.20.30/24;
    }
  }
}
```

The following example shows the contents of the **cust1_accounting** file in the **/var/log** folder that might result from the preceding configuration:

```
#FILE CREATED 995495212 2001-07-18-22:26:52
#hostname host
#profile-layout cust1_profile,epoch-timestamp,interfaces,filter-name,
counter-name,packet-count,byte-count
cust1_profile,995495572,xe-1/0/0.0,f3-xe-1/0/0.0-i,r1-xe-1/0/0.0-i,5953,1008257
cust1_profile,995495602,xe-1/0/0.0,f3-xe-1/0/0.0-o,r1-xe-1/0/0.0-o,5929,1006481
...
```

If the **interface-specific** statement is not included in the configuration, the following output might result:

```
#FILE CREATED 995495212 2001-07-18-22:26:52
#hostname host
#profile-layout cust1_profile,epoch-timestamp,interfaces,filter-name,
counter-name,packet-count,byte-count
cust1_profile,995495572,xe-1/0/0.0,f3,r1,5953,1008257
cust1_profile,995495632,xe-1/0/0.0,f3,r1,5929,1006481
```

Related Documentation

- Configuring the Filter Profile on page 177
- Configuring the Interface Profile on page 175

Configuring SCU or DCU

To configure SCU or DCU, perform the following tasks described in this section:

NOTE: We recommend that you stop the network traffic on an interface before you modify the DCU or SCU configuration for that interface. Modifying the DCU or SCU configuration without stopping the traffic might corrupt the DCU or SCU statistics. Before you restart the traffic after modifying the configuration, enter the **clear interfaces statistics** command.

- Creating Prefix Route Filters in a Policy Statement on page 182
- Applying the Policy to the Forwarding Table on page 182
- Enabling Accounting on Inbound and Outbound Interfaces on page 182

Creating Prefix Route Filters in a Policy Statement

To define prefix router filters:

```
[edit policy-options]
policy-statement scu-1 {
   term term1;
   from {
      route-filter 192.168.1.0/24 orlonger;
   }
   then source-class gold;
}
```

Applying the Policy to the Forwarding Table

To apply the policy to the forwarding table:

```
[edit]
routing-options {
   forwarding-table {
      export scu-1;
   }
}
```

Enabling Accounting on Inbound and Outbound Interfaces

To enable accounting on inbound and outbound interfaces:

```
[edit]
interfaces {
   so-6/1/0 {
      unit 0 {
         family inet;
         accounting {
            destination-class-usage;
            source-class-usage {
               output;
            }
         }
      }
   }
}
[edit]
```

```
interfaces {
  xe-0/1/0 {
    unit 0 {
      family inet6 {
        accounting {
          source-class-usage {
            input;
          }
        }
      }
    }
  }
}
```

Optionally, you can include the input and output statements on a single interface as shown:

```
[edit]
interfaces {
  xe-0/1/2 {
    unit 0 {
      family inet6 {
        accounting {
          source-class-usage {
            input;
            output;
          }
        }
      }
    }
  }
}
```

For more information about configuring route filters and source classes in a routing policy, see the *Routing Policy Feature Guide for Routing Devices* and the *Junos OS Network Interfaces Library for Routing Devices*.

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 82
- Configuring SCU on a Virtual Loopback Tunnel Interface on page 184
- Configuring Class Usage Profiles on page 185
- Configuring the MIB Profile on page 187
- Configuring the Routing Engine Profile on page 189

Configuring SCU on a Virtual Loopback Tunnel Interface

To configure source class usage on the virtual loopback tunnel interface, perform the tasks described in the following sections:

Example: Configuring a Virtual Loopback Tunnel Interface on a Provider Edge Router Equipped with a Tunnel PIC

Define a virtual loop interface on a provider edge router with a Tunnel PIC:

```
[edit interfaces]
vt-0/3/0 {
   unit 0 {
      family inet {
         accounting {
            source-class-usage {
               input;
            }
         }
      }
   }
}
```

Example: Mapping the VRF Instance Type to the Virtual Loopback Tunnel Interface

Map the VRF instance type to the virtual loopback tunnel interface:

```
[edit]
routing-instances {
   VPN-A {
      instance-type vrf;
      interface at-2/1/1.0;
      interface vt-0/3/0.0;
      route-distinguisher 10.255.14.225:100;
      vrf-import import-policy-name;
      vrf-export export-policy-name;
      protocols {
         bgp {
            group to-r4 {
               local-address 10.27.253.1;
               peer-as 400;
               neighbor 10.27.253.2;
            }
         }
      }
   }
```

```
}
```

NOTE: For SCU and DCU to work, do not include the **vrf-table-label** statement at the **[edit routing-instances *instance-name*]** hierarchy level.

Example: Sending Traffic Received from the Virtual Loopback Interface Out the Source Class Output Interface

Send traffic received from the virtual loopback tunnel interface out of the source class output interface:

```
[edit interfaces]
at-1/1/0 {
  unit 0 {
    family inet {
      accounting {
        source-class-usage {
          output;
        }
      }
    }
  }
}
```

For more information about configuring source class usage on the virtual loopback tunnel interface, see the *Junos OS Network Interfaces Library for Routing Devices*.

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 82
- Configuring SCU or DCU on page 181
- Configuring Class Usage Profiles on page 185
- Configuring the MIB Profile on page 187
- Configuring the Routing Engine Profile on page 189

Configuring Class Usage Profiles

To collect class usage statistics, perform the tasks described in these sections:

- Configuring a Class Usage Profile on page 185
- Configuring the File Information on page 186
- Configuring the Interval on page 186
- Creating a Class Usage Profile to Collect Source Class Usage Statistics on page 186
- Creating a Class Usage Profile to Collect Destination Class Usage Statistics on page 187

Configuring a Class Usage Profile

You can configure the class usage profile to collect statistics for particular source and destination classes.

To configure the class usage profile to filter by source classes, include the **source-classes** statement at the **[edit accounting-options class-usage-profile *profile-name*]** hierarchy level:

```
[edit accounting-options class-usage-profile profile-name]
source-classes {
  source-class-name;
}
```

To configure the class usage profile to filter by destination classes, include the **destination-classes** statement at the **[edit accounting-options class-usage-profile *profile-name*]** hierarchy level:

```
[edit accounting-options class-usage-profile profile-name]
destination-classes {
  destination-class-name;
}
```

Configuring the File Information

Each accounting profile logs its statistics to a file in the **/var/log** directory.

To specify which file to use, include the **file** statement at the **[edit accounting-options class-usage-profile *profile-name*]** hierarchy level:

```
[edit accounting-options class-usage-profile profile-name]
file filename;
```

You must specify a filename for the source class usage profile that has already been configured at the **[edit accounting-options]** hierarchy level. You can also specify a filename for the destination class usage profile configured at the **[edit accounting-options]** hierarchy level.

Configuring the Interval

Each interface with a class usage profile enabled has statistics collected once per interval specified for the accounting profile. Statistics collection time is scheduled evenly over the configured interval. To configure the interval, include the **interval** statement at the **[edit accounting-options class-usage-profile *profile-name*]** hierarchy level:

```
[edit accounting-options class-usage-profile profile-name]
interval;
```

Creating a Class Usage Profile to Collect Source Class Usage Statistics

To create a class usage profile to collect source class usage statistics:

```
[edit]
accounting-options {
  class-usage-profile scu-profile1;
  file usage-stats;
  interval 15;
  source-classes {
    gold;
    silver;
    bronze;
  }
}
```

The class usage profile, **scu-profile1**, writes data to the file **usage_stats**. The file might look like the following:

```
#FILE CREATED 976825278 2000-12-14-20:21:18
#profile-layout, scu_profile,epoch-timestamp,interface-name,source-class,
packet-count,byte-count
scu_profile,980313078,xe-1/0/0.0,gold,82,6888
scu_profile,980313078,xe-1/0/0.0,silver,164,13776
scu_profile,980313078,xe-1/0/0.0,bronze,0,0
scu_profile,980313678,xe-1/0/0.0,gold,82,6888
scu_profile,980313678,xe-1/0/0.0,silver,246,20664
scu_profile,980313678,xe-1/0/0.0,bronze,0,0
```

Creating a Class Usage Profile to Collect Destination Class Usage Statistics

To create a class usage profile to collect destination class usage statistics:

```
[edit]
accounting-options {
   class-usage-profile dcu-profile1;
   file usage-stats
   interval 15;
   destination-classes {
      gold;
      silver;
      bronze;
   }
}
```

The class usage profile, **dcu-profile1**, writes data to the file **usage-stats**. The file might look like the following:

```
#FILE CREATED 976825278 2000-12-14-20:21:18
#profile-layout, dcu_profile,epoch-timestamp,interface-name,destination-class,
packet-count,byte-count
dcu_profile,980313078,xe-1/0/0.0,gold,82,6888
dcu_profile,980313078,xe-1/0/0.0,silver,164,13776
dcu_profile,980313078,xe-1/0/0.0,bronze,0,0
dcu_profile,980313678,xe-1/0/0.0,gold,82,6888
dcu_profile,980313678,xe-1/0/0.0,silver,246,20664
dcu_profile,980313678,xe-1/0/0.0,bronze,0,0
...
#FILE CLOSED 976826178 2000-12-14-20:36:18
```

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 82
- Configuring SCU or DCU on page 181
- Configuring SCU on a Virtual Loopback Tunnel Interface on page 184
- Configuring the Routing Engine Profile on page 189

Configuring the MIB Profile

The MIB profile collects MIB statistics and logs them to a file. The MIB profile specifies the SNMP operation and MIB object names for which statistics are collected.

To configure a MIB profile, include the **mib-profile** statement at the **[edit accounting-options]** hierarchy level:

```
[edit accounting-options]
mib-profile profile-name {
  file filename;
  interval minutes;
  object-names {
    mib-object-name;
  }
  operation operation-name;
}
```

To configure a MIB profile, perform the tasks described in the following sections:

Configuring the File Information

Each accounting profile logs its statistics to a file in the **/var/log** directory.

To configure which file to use, include the **file** statement at the **[edit accounting-options mib-profile *profile-name*]** hierarchy level:

```
[edit accounting-options mib-profile profile-name]
file filename;
```

You must specify a ***filename*** for the MIB profile that has already been configured at the **[edit accounting-options]** hierarchy level.

Configuring the Interval

A MIB profile has statistics collected once per interval time specified for the profile. Statistics collection time is scheduled evenly over the configured interval. To configure the interval, include the **interval** statement at the **[edit accounting-options mib-profile *profile-name*]** hierarchy level:

```
[edit accounting-options mib-profile profile-name]
interval;
```

The range for the **interval** statement is 1 through 2880 minutes. The default is 30 minutes.

Configuring the MIB Operation

A MIB profile must specify the operation that is used to collect MIB statistics. To configure which operation is used to collect MIB statistics, include the **operation** statement at the **[edit accounting-options mib-profile *profile-name*]** hierarchy level:

```
[edit accounting-options mib-profile profile-name]
operation operation-name;
```

You can configure a **get**, **get-next**, or **walk** operation. The default operation is **walk**.

Configuring MIB Object Names

A MIB profile must specify the MIB objects for which statistics are to be collected. To configure the MIB objects for which statistics are collected, include the **objects-names** statement at the **[edit accounting-options mib-profile *profile-name*]** hierarchy level:

```
[edit accounting-options mib-profile profile-name]
object-names {
  mib-object-name;
}
```

You can include multiple MIB object names in the configuration.

NOTE: Do not configure MIB objects related to interface octets or packets for a MIB profile, because it can cause the SNMP walk or a CLI show command to time out.

Example: Configuring a MIB Profile

Configure a MIB profile:

```
[edit accounting-options]
mib-profile mstatistics {
  file stats;
  interval 60;
  operation walk;
  objects-names {
    ipCidrRouteStatus;
  }
}
```

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 82
- Configuring SCU or DCU on page 181
- Configuring SCU on a Virtual Loopback Tunnel Interface on page 184
- Configuring Class Usage Profiles on page 185
- Configuring the Routing Engine Profile on page 189

Configuring the Routing Engine Profile

The Routing Engine profile collects Routing Engine statistics and logs them to a file. The Routing Engine profile specifies the fields for which statistics are collected.

To configure a Routing Engine profile, include the **routing-engine-profile** statement at the **[edit accounting-options]** hierarchy level:

```
[edit accounting-options]
routing-engine-profile profile-name {
  fields {
    field-name;
```

```
    }
    file filename;
    interval minutes;
}
```

To configure a Routing Engine profile, perform the tasks described in the following sections:

- Configuring Fields on page 190
- Configuring the File Information on page 190
- Configuring the Interval on page 190
- Example: Configuring a Routing Engine Profile on page 190

Configuring Fields

A Routing Engine profile must specify what statistics are collected. To configure which statistics should be collected for the Routing Engine, include the **fields** statement at the **[edit accounting-options routing-engine-profile *profile-name*]** hierarchy level:

```
[edit accounting-options routing-engine-profile profile-name]
fields {
    field-name;
}
```

Configuring the File Information

Each accounting profile logs its statistics to a file in the **/var/log** directory.

To configure which file to use, include the **file** statement at the **[edit accounting-options routing-engine-profile *profile-name*]** hierarchy level:

```
[edit accounting-options routing-engine-profile profile-name]
file filename;
```

You must specify a ***filename*** for the Routing Engine profile that has already been configured at the **[edit accounting-options]** hierarchy level.

Configuring the Interval

A Routing Engine profile has statistics collected once per interval time specified for the profile. Statistics collection time is scheduled evenly over the configured interval. To configure the interval, include the **interval** statement at the **[edit accounting-options routing-engine-profile *profile-name*]** hierarchy level:

```
[edit accounting-options routing-engine-profile profile-name]
interval;
```

The range for **interval** is 1 through 2880 minutes. The default is 30 minutes.

Example: Configuring a Routing Engine Profile

Configure a Routing Engine profile:

```
[edit accounting-options]
file my-file {
    size 300k;
}
```

```
routing-engine-profile profile-1 {
  file my-file;
  fields {
    host-name;
    date;
    time-of-day;
    uptime;
    cpu-load-1;
    cpu-load-5;
    cpu-load-15;
  }
}
```

Related Documentation

- Understanding Source Class Usage and Destination Class Usage Options on page 82
- Configuring SCU or DCU on page 181
- Configuring SCU on a Virtual Loopback Tunnel Interface on page 184
- Configuring Class Usage Profiles on page 185
- Configuring the MIB Profile on page 187

CHAPTER 11

SNMP Configuration Statements

access-list

Syntax

```
[edit snmp]
  routing-instance-access {
    access-list {
      routing-instance;
      routing-instance restrict;
    }
  }
```

Hierarchy Level [edit snmp routing-instance-access]

Release Information Statement introduced in Junos OS Release 8.4.

Description Create access lists to control SNMP agents in routing instances from accessing SNMP information. To enable the SNMP agent on a routing instance to access SNMP information, specify the routing instance name. To disable the SNMP agent on a routing instance from accessing SNMP information, include the routing-instance name followed by the **restrict** keyword.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- routing-instance-access on page 209

agent-address

Syntax agent-address outgoing-interface;

Hierarchy Level [edit snmp trap-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Set the agent address of all SNMPv1 traps generated by this router or switch. Currently, the only option is **outgoing-interface**, which sets the agent address of each SNMPv1 trap to the address of the outgoing interface of that trap.

Options **outgoing-interface**—Value of the agent address of all SNMPv1 traps generated by this router or switch. The **outgoing-interface** option sets the agent address of each SNMPv1 trap to the address of the outgoing interface of that trap.

Default: disabled (the agent address is not specified in SNMPv1 traps).

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Agent Address for SNMP Traps on page 103

authorization

Syntax authorization *authorization*;

Hierarchy Level [edit snmp community *community-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Set the access authorization for SNMP **Get**, **GetBulk**, **GetNext**, and **Set** requests.

Options ***authorization***—Access authorization level:

- **read-only**—Enable **Get**, **GetNext**, and **GetBulk** requests.
- **read-write**—Enable all requests, including **Set** requests. You must configure a view to enable **Set** requests.

Default: **read-only**

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the SNMP Community String on page 95

categories

Syntax
```
categories {
  category;
}
```

Hierarchy Level [edit snmp trap-group *group-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define the types of traps that are sent to the targets of the named trap group.

Default If you omit the **categories** statement, all trap types are included in trap notifications.

Options *category*—Name of a trap type: **authentication**, **chassis**, **configuration**, **link**, **remote-operations**, **rmon-alarm**, **routing**, **sonet-alarms**, **startup**, or **vrrp-events**.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Groups on page 104

client-list

Syntax
```
client-list client-list-name {
  ip-addresses;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.5.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for QFX Series switches.

Description Define a list of SNMP clients.

Options ***client-list-name***—Name of the client list.

ip-addresses—IP addresses of the SNMP clients to be added to the client list,

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Adding a Group of Clients to an SNMP Community on page 96

client-list-name

Syntax client-list-name *client-list-name*;

Hierarchy Level [edit snmp community *community-name*]

Release Information Statement introduced in Junos OS Release 8.5.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for FX Series switches.

Description Add a client list or prefix list to an SNMP community.

Options ***client-list-name***—Name of the client list or prefix list.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Adding a Group of Clients to an SNMP Community on page 96

clients

Syntax
```
clients {
  address <restrict>;
}
```

Hierarchy Level [edit snmp community *community-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for FX Series switches.

Description Specify the IPv4 or IPv6 addresses of the SNMP client hosts that are authorized to use this community.

Default If you omit the **clients** statement, all SNMP clients using this community string are authorized to access the router.

Options ***address***—Address of an SNMP client that is authorized to access this router. You must specify an address, not a hostname. To specify more than one client, include multiple ***address*** options.

restrict—(Optional) Do not allow the specified SNMP client to access the router.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the SNMP Community String on page 95

commit-delay

Syntax commit-delay *seconds*;

Hierarchy Level [edit snmp nonvolatile]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure the timer for the SNMP **Set** reply and start of the commit.

Options ***seconds***—Delay between an affirmative SNMP **Set** reply and start of the commit.

Default: 5 seconds

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Commit Delay Timer on page 100

community (SNMP)

Syntax

```
community community-name {
  authorization authorization;
  client-list-name client-list-name;
  clients {
    address restrict;
  }
  view view-name;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define an SNMP community. An SNMP community authorizes SNMP clients based on the source IP address of incoming SNMP request packets. A community also defines which MIB objects are available and the operations (read-only or read-write) allowed on those objects.

The SNMP client application specifies an SNMP community name in **Get**, **GetBulk**, **GetNext**, and **Set** SNMP requests.

Default If you omit the **community** statement, all SNMP requests are denied.

Options *community-name*—Community string. If the name includes spaces, enclose it in quotation marks (" ").

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the SNMP Community String on page 95

contact (SNMP)

Syntax contact *contact*;

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define the value of the MIB II **sysContact** object, which is the contact person for the managed system.

Options ***contact***—Name of the contact person. If the name includes spaces, enclose it in quotation marks (" ").

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the System Contact on a Device Running Junos OS on page 93

description (SNMP)

Syntax description *description*;

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Define the value of the MIB II **sysDescription** object, which is the description of the system being managed.

Options ***description***—System description. If the name includes spaces, enclose it in quotation marks (" ").

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the System Description on a Device Running Junos OS on page 94

destination-port

Syntax destination-port *port-number*;

Hierarchy Level [edit snmp trap-group]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Assign a trap port number other than the default.

Default If you omit this statement, the default port is 162.

Options ***port-number***—SNMP trap port number.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Groups on page 104

enterprise-oid

Syntax enterprise-oid;

Hierarchy Level [edit snmp trap-options]

Release Information Statement introduced in Junos OS Release 10.0

Description Add the **snmpTrapEnterprise** object, which shows the association between an enterprise-specific trap and the organization that defined the trap, to standard SNMP traps. By default, the **snmpTrapEnterprise** object is added only to the enterprise-specific traps. When the **enterprise-oid** statement is included in the configuration, **snmpTrapEnterprise** is added to all the traps generated from the device.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Options on page 101

filter-duplicates

Syntax filter-duplicates;

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Filter duplicate **Get**, **GetNext**, or **GetBulk** SNMP requests.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Filtering Duplicate SNMP Requests on page 99

filter-interfaces

Syntax

```
filter-interfaces {
  interfaces {
    all-internal-interfaces;
    interface 1;
    interface 2;
  }
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 9.4.
Statement introduced in Junos OS Release 9.4 for EX Series Switches.

Description Filter out information related to specific interfaces from the output of SNMP **Get** and **GetNext** requests performed on interface-related MIBs.

Options

all-internal-interfaces—Filters out information from SNMP **Get** and **GetNext** requests for the specified interfaces.

interfaces—Specifies the interfaces to filter out from the output of SNMP **Get** and **GetNext** requests.

NOTE: Starting with Release 12.1, Junos OS provides an except option (! operator) that enables you to filter out all interfaces except those interfaces that match the regular expressions prefixed with the ! mark.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Filtering Interface Information Out of SNMP Get and GetNext Output on page 108

location (SNMP)

Syntax location *location*;

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define the value of the MIB II **sysLocation** object, which is the physical location of the managed system.

Options ***location***—Location of the local system. You must enclose the name within quotation marks (" ").

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the System Location for a Device Running Junos OS on page 93

logical-system (SNMP

Syntax

```
logical-system logical-system-name {
  routing-instance routing-instance-name;
}
```

Hierarchy Level

[edit snmp community *community-name*],
[edit snmp trap-group],
[edit snmp trap-options]
[edit snmp v3target-address *target-address-name*]

Release Information

Statement introduced in Junos OS Release 9.3
Statement introduced in Junos OS Release 9.0 for EX Series switches.

NOTE: The **logical-system** statement replaces the **logical-router** statement, and is backward-compatible with Junos OS Release 8.3 and later.

Description

Specify a logical system name for SNMP v1 and v2c clients.

Include at the **[edit snmp trap-options]** hierarchy level to specify a logical-system address as the source address of an SNMP trap.

Include at the **[edit snmp v3 target-address]** hierarchy level to specify a logical-system name as the destination address for an SNMPv3 trap or inform.

Options

logical-system-name—Name of the logical system.

routing-instance ***routing-instance-name***—Statement to specify a routing instance associated with the logical system.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152
- Configuring the Trap Target Address on page 133

logical-system-trap-filter

Syntax logical-system-trap-filter;

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.4.

Description Restrict the routing instances from receiving traps that are not related to the logical system networks to which they belong.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Trap Support for Routing Instances on page 18

name

Syntax name *name*;

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Set the system name from the command-line interface.

Options *name*—System name override.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the System Name on page 94

nonvolatile

Syntax

```
nonvolatile {
  commit-delay seconds;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
The commit-delay statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure options for SNMP **Set** requests.

The statement is explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Commit Delay Timer on page 100
- commit-delay on page 197

oid

Syntax oid *object-identifier* (exclude | include);

Hierarchy Level [edit snmp view *view-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify an object identifier (OID) used to represent a subtree of MIB objects.

Options **exclude**—Exclude the subtree of MIB objects represented by the specified OID.

include—Include the subtree of MIB objects represented by the specified OID.

object-identifier—OID used to represent a subtree of MIB objects. All MIB objects represented by this statement have the specified OID as a prefix. You can specify the OID using either a sequence of dotted integers or a subtree name.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring MIB Views on page 109

proxy (snmp)

Syntax

```
proxy proxy-name{
  device-name device-name;
  logical-system logical-system {
    routing-instance routing-instance;
  }
  routing-instance routing-instance;
  (version-v1 | version-v2c) {
    snmp-community community-name;
    no-default-comm-to-v3-config;
  }
  version-v3 {
    security-name security-name;
    context context-name;
  }
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 12.3.

Description Configure a device to act as a proxy SNMP agent, and specify a name for the proxy.

Options

context *context-name*—Specify the SNMPv3 context name as configured on the device specified at **edit snmp proxy *proxy-name* device-name *device-name***.

For more information about this statement, see *context*.

device-name *device-name*—Specify the name of the device to be managed through the proxy SNMP agent.

no-default-comm-to-v3-config—(Optional) Specify whether you have to manually configure the statements at the **[edit snmp v3 snmp-community *community-name*]** and **[edit snmp v3 vacm]** hierarchy levels.

If this statement is not included in the configuration, the **[edit snmp v3 snmp-community *community-name*]** and **[edit snmp v3 vacm]** hierarchy level configurations are automatically initialized.

proxy-name—Specify the name of the proxy.

security-name *security-name*—Specify the SNMPv3 security name as configured on the device specified at **edit snmp proxy *proxy-name* device-name *device-name***.

For more information about this statement, see security-name.

snmp-community *community-name*—Specify the name of the SNMP community. The community name you configure should match the **snmp-community** configuration on the device specified at **edit snmp proxy *proxy-name* device-name *device-name***. For more information about this statement, see snmp-community.

(version-v1 | version-v2c)—Specify the SNMP version, and add the relevant configuration.

version-v3—Add the SNMPv3 configuration.

The remaining statements are explained separately.

Required Privilege Level
snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring a Proxy SNMP Agent on page 98

routing-instance (SNMP)

Syntax
routing-instance *routing-instance-name*;

Hierarchy Level
[edit snmp community *community-name*],
[edit snmp community *community-name* logical-system *logical-system-name*],
[edit snmp trap-group *group*]

Release Information
Statement introduced in Junos OS Release 8.3.
Added to the **[edit snmp community *community-name*]** hierarchy level in Junos OS Release 8.4.
Added to the **[edit snmp community *community-name* logical-system *logical-system-name*]** hierarchy level in Junos OS Release 9.1.
Statement introduced in Junos OS Release 9.1 for EX Series switches.

Description
Specify a routing instance for SNMPv1 and SNMPv2 trap targets. All targets configured in the trap group use this routing instance.

If the routing instance is defined within a logical system, include the **logical-system *logical-system-name*** statement at the **[edit snmp community *community-name*]** hierarchy level and specify the **routing-instance** statement under the **[edit snmp community *community-name* logical-system *logical system-name*]** hierarchy level.

Options
routing-instance-name—Name of the routing instance.

Required Privilege Level
snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Groups on page 104
- Configuring the Source Address for SNMP Traps on page 102
- Specifying a Routing Instance in an SNMPv1 or SNMPv2c Community on page 152

routing-instance-access

Syntax

```
[edit snmp]
  routing-instance-access {
    access-list {
      routing-instance;
      routing-instance restrict;
    }
  }
```

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.4.

Description Enable SNMP managers in routing instances other than the default routing instance to access SNMP information. For information about the **access-list** option, see access-list.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Enabling SNMP Access over Routing Instances on page 152

snmp

Syntax snmp { ... }

Hierarchy Level [edit]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure SNMP.

SNMP modules cannot have the slash (/) character or the @ character in the name.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91

source-address (SNMP)

Syntax source-address *address*;

Hierarchy Level [edit snmp trap-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Set the source address of every SNMP trap packet sent by this router to a single address regardless of the outgoing interface. If the source address is not specified, the default is to use the address of the outgoing interface as the source address.

Options ***address***—Source address of SNMP traps. You can configure the source address of trap packets two ways: **lo0** or a valid IPv4 address configured on one of the router interfaces. The value **lo0** indicates that the source address of all SNMP trap packets is set to the lowest loopback address configured at interface **lo0**.

Default: Disabled. (The source address is the address of the outgoing interface.)

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Source Address for SNMP Traps on page 102

targets

Syntax
```
targets {
  address;
}
```

Hierarchy Level [edit snmp trap-group *group-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure one or more systems to receive SNMP traps.

Options ***address***—IPv4 or IPv6 address of the system to receive traps. You must specify an address, not a hostname.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Groups on page 104

traceoptions (SNMP)

Syntax

```
traceoptions {
  file filename <files number> <match regular-expression> <size size> <world-readable |
    no-world-readable>;
  flag flag;
  no-remote-trace;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
file *filename* option added in Junos OS Release 8.1.
world-readable | no-world-readable option added in Junos OS Release 8.1.
match *regular-expression* option added in Junos OS Release 8.1.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The output of the tracing operations is placed into log files in the **/var/log** directory. Each log file is named after the SNMP agent that generates it. Currently, the following logs are created in the **/var/log** directory when the **traceoptions** statement is used:

- chassisd
- craftd
- ilmid
- mib2d
- rmopd
- serviced
- snmpd

Options

file *filename*—By default, the name of the log file that records trace output is the name of the process being traced (for example, **mib2d** or **snmpd**). Use this option to specify another name.

files *number*—(Optional) Maximum number of trace files per SNMP subagent. When a trace file (for example, **snmpd**) reaches its maximum size, it is archived by being renamed to **snmpd.0**. The previous **snmpd.1** is renamed to **snmpd.2**, and so on. The oldest archived file is deleted.

Range: 2 through 1000 files

Default: 10 files

flag *flag*—Tracing operation to perform. To specify more than one tracing operation, include multiple **flag** statements:

- **all**—Log all SNMP events.
- **configuration**—Log reading of configuration at the **[edit snmp]** hierarchy level.

- **database**—Log events involving storage and retrieval in the events database.
- **events**—Log important events.
- **general**—Log general events.
- **interface-stats**—Log physical and logical interface statistics.
- **nonvolatile-sets**—Log nonvolatile SNMP set request handling.
- **pdu**—Log SNMP request and response packets.
- **policy**—Log policy processing.
- **protocol-timeouts**—Log SNMP response timeouts.
- **routing-socket**—Log routing socket calls.
- **server**—Log communication with processes that are generating events.
- **subagent**—Log subagent restarts.
- **timer-events**—Log internally generated events.
- **varbind-error**—Log variable binding errors.

match *regular-expression*—(Optional) Refine the output to include lines that contain the regular expression.

size *size*—(Optional) Maximum size, in kilobytes (KB), of each trace file before it is closed and archived.

Range: 10 KB through 1 GB

Default: 1000 KB

world-readable | no-world-readable—(Optional) By default, log files can be accessed only by the user who configures the tracing operation. The **world-readable** option enables any user to read the file. To explicitly set the default behavior, use the **no-world-readable** option.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Tracing SNMP Activity on a Device Running Junos OS on page 299

trap-group

Syntax

```
trap-group group-name {
  categories {
    category;
  }
  destination-port port-number;
  routing-instance instance;
  targets {
    address;
  }
  version (all | v1 | v2);
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a named group of hosts to receive the specified trap notifications. The name of the trap group is embedded in SNMP trap notification packets as one variable binding (varbind) known as the community name. At least one trap group must be configured for SNMP traps to be sent.

Options *group-name*—Name of the trap group. If the name includes spaces, enclose it in quotation marks (" ").

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring SNMP Trap Groups on page 104

trap-options

Syntax

```
trap-options {
    agent-address outgoing-interface;
    source-address address;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Using SNMP trap options, you can set the source address of every SNMP trap packet sent by the router or switch to a single address, regardless of the outgoing interface. In addition, you can set the agent address of each SNMPv1 trap. For more information about the contents of SNMPv1 traps, see RFC 1157.

The remaining statements are explained separately.

Default Disabled

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring SNMP Trap Options on page 101

version (SNMP)

Syntax version (all | v1 | v2);

Hierarchy Level [edit snmp trap-group *group-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the version number of SNMP traps.

Default **all**—Send an SNMPv1 and SNMPv2 trap for every trap condition.

Options **all**—Send an SNMPv1 and SNMPv2 trap for every trap condition.

v1—Send SNMPv1 traps only.

v2—Send SNMPv2 traps only.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Trap Groups on page 104

view (Associating a MIB View with a Community)

Syntax view *view-name*;

Hierarchy Level [edit snmp community *community-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Associate a view with a community. A view represents a group of MIB objects.

Options ***view-name***—Name of the view. You must use a view name already configured in the **view** statement at the **[edit snmp]** hierarchy level.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the SNMP Community String on page 95

view (Configuring a MIB View)

Syntax

```
view view-name {
  oid object-identifier (include | exclude);
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Define a MIB view. A MIB view identifies a group of MIB objects. Each MIB object in a view has a common OID prefix. Each object identifier represents a subtree of the MIB object hierarchy. The **view** statement uses a view to specify a group of MIB objects on which to define access. To enable a view, you must associate the view with a community by including the **view** statement at the **[edit snmp community *community-name*]** hierarchy level.

NOTE: To remove an OID completely, use the **delete view all oid oid-number** command but omit the **include** parameter.

Options *view-name*—Name of the view.

The remaining statement is explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring MIB Views on page 109
- Associating MIB Views with an SNMP User Group on page 125
- **community on page 198**

CHAPTER 12

SNMPv3 Configuration Statements

address (SNMP)

Syntax address *address*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the SNMP target address.

Options ***address***—IPv4 address of the system to receive traps or informs. You must specify an address, not a hostname.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Address on page 134

address-mask

Syntax address-mask *address-mask*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 on the QFX Series.

Description Define and verify the source addresses for a group of target addresses for SNMP traps and informs.

Options ***address-mask***—Define a range of addresses.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Address Mask on page 134

authentication-md5

Syntax
```
authentication-md5 {
    authentication-password authentication-password;
}
```

Hierarchy Level [edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user username]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure MD5 as the authentication type for the SNMPv3 user.

NOTE: You can only configure one authentication type for each SNMPv3 user.

The remaining statement is explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring MD5 Authentication on page 119

authentication-none

Syntax authentication-none;

Hierarchy Level [edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure that there should be no authentication for the SNMPv3 user.

NOTE: You can configure only one authentication type for each SNMPv3 user.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring No Authentication on page 120

authentication-password

Syntax authentication-password *authentication-password*;

Hierarchy Level [edit snmp v3 usm local-engine user *username* authentication-md5],
[edit snmp v3 usm local-engine user *username* authentication-sha],
[edit snmp v3 usm remote-engine *engine-id* user *username* authentication-md5],
[edit snmp v3 usm remote-engine *engine-id* user *username* authentication-sha]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the password for user authentication.

Options ***authentication-password***—Password that a user enters. The password is then converted into a key that is used for authentication.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include lowercase letters, uppercase letters, numbers, and the following special characters:

  , . / \ < > ; : ' [ ] { } ~ ! @ # $ % ^ * _ + = - ` ?

  In addition, the following four special characters are also supported, but you must enclose them within quotation marks (“”) if you enter them on the CLI; if you use a Network Management System to enter the password, the quotation marks are not required:

  | & ( )

  Control characters—entered by simultaneously pressing the Ctrl key and additional keys—are not supported.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring MD5 Authentication on page 119
- Configuring SHA Authentication on page 119

authentication-sha

Syntax

```
authentication-sha {
    authentication-password authentication-password;
}
```

Hierarchy Level

[edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description

Configure the secure hash algorithm (SHA) as the authentication type for the SNMPv3 user.

NOTE: You can configure only one authentication type for each SNMPv3 user.

The remaining statement is explained separately.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring SHA Authentication on page 119

community-name (SNMP)

Syntax community-name *community-name*;

Hierarchy Level [edit snmp v3 snmp-community *community-index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11. for the QFX Series.

Description Define an SNMP community to authorize SNMPv1 or SNMPv2c clients in an SNMPv3 system. When you configure a community in SNMPv3, you can also specify a security name. The access privileges associated with the security name determine which MIB objects are available and which operations (read, write, or notify) are allowed on those objects.

Options ***community-name***—Community string for an SNMPv1 or SNMPv2c community. If unconfigured, it is the same as the community index. If the name includes spaces, enclose the name in quotation marks (" ").

NOTE: Community names must be unique. You cannot configure the same community name at the **[edit snmp community]** and **[edit snmp v3 snmp-community *community-index*]** hierarchy levels.

The community name at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level is encrypted and not displayed in the command-line interface (CLI).

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the SNMPv3 Community on page 144

engine-id (SNMP)

Syntax

```
engine-id {
    (local engine-id-suffix | use-default-ip-address | use-mac-address);
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.1 for EX Series switches.

Description The local engine ID is defined as the administratively unique identifier of an SNMPv3 engine, and is used for identification, not for addressing. There are two parts of an engine ID: prefix and suffix. The prefix is formatted according to the specifications defined in RFC 3411, *An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks*. You can configure the suffix here.

NOTE: SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID. If you configure or change the engine ID, you must commit the new engine ID before you configure SNMPv3 users. Otherwise the keys generated from the configured passwords are based on the previous engine ID.

For the engine ID, we recommend using the MAC address of the management port.

Options

local *engine-id-suffix*—Explicit setting for the engine ID suffix.

use-default-ip-address—The engine ID suffix is generated from the default IP address.

use-mac-address—The SNMP engine identifier is generated from the MAC address of the management interface on the router.

Default: use-default-ip-address

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Local Engine ID on page 111

group (Configuring Access Privileges)

Syntax

```
group group-name {
  (default-context-prefix | context-prefix context-prefiix){
    security-model (any | usm | v1 | v2c) {
      security-level (authentication | none | privacy) {
        notify-view view-name;
        read-view view-name;
        write-view view-name;
      }
    }
  }
}
```

Hierarchy Level [edit snmp v3 vacm access]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Assign the security name to a group, and specify the SNMPv3 context applicable to the group. The **default-context-prefix** statement, when included, adds all the contexts configured on the device to the group, whereas the **context-prefix *context-prefix*** statement enables you to specify a context and to add that particular context to the group.

(Not applicable to the QFX Series.) When the context prefix is specified as default (for example, **context-prefix default**), the context associated with the master routing instance is added to the group. To specify a routing instance that is part of a logical system, specify it as **logical system/routing instance**. For example, to specify routing instance ri1 in logical system ls1, include **context-prefix ls1/ri1**.

The remaining statements under this hierarchy are explained separately.

Options ***group-name***—SNMPv3 group name created for the SNMPv3 group.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Group on page 124

group (Associating a Security Name)

Syntax group *group-name*;

Hierarchy Level [edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c) security-name *security-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Associate a security name with a group composed of users with the same access privileges. The security name is used during authentication of SNMP messages, and is mapped to a username.

Options ***group-name***—Collection of SNMP security names that share the same SNMPv3 access privileges.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Group on page 128

retry-count (SNMPv3)

Syntax retry-count *number*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced in Junos OS Release 7.4.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the retry count for SNMP informs.

Options ***number***—Maximum number of times the inform is transmitted if no acknowledgment is received. If no acknowledgment is received after the inform is transmitted the maximum number of times, the inform message is discarded.

Default: 3 times

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Informs on page 139
- timeout on page 226

timeout (SNMP)

Syntax timeout *seconds*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced in Junos OS Release 7.4.

Description Configure the timeout period (in seconds) for SNMP informs.

Options ***seconds***—Number of seconds to wait for an inform acknowledgment. If no acknowledgment is received within the timeout period, the inform is retransmitted.

Default: 15

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring SNMP Informs on page 139
- retry-count (SNMPv3) on page 225

local-engine

Syntax

```
local-engine {
  user username {
    authentication-md5 {
      authentication-password authentication-password;
    }
    authentication-none;
    authentication-sha {
      authentication-password authentication-password;
    }
    privacy-aes128 {
      privacy-password privacy-password;
    }
    privacy-des {
      privacy-password privacy-password;
    }
    privacy-3des {
      privacy-password privacy-password;
    }
    privacy-none {
      privacy-password privacy-password;
    }
  }
}
```

Hierarchy Level [edit snmp v3 usm]

Release Information
Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure local engine information for the user-based security model (USM).

The remaining statements are explained separately.

Required Privilege Level
snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Creating SNMPv3 Users on page 118

message-processing-model

Syntax message-processing-model (v1 | v2c | v3);

Hierarchy Level [edit snmp v3 target-parameters *target-parameter-name* parameters]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the message processing model to be used when generating SNMP notifications.

Options

v1—SNMPv1 message process model.

v2c—SNMPv2c message process model.

v3—SNMPv3 message process model.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Message Processing Model on page 138

notify

Syntax

```
notify name {
  tag tag-name;
  type (trap | inform);
}
```

Hierarchy Level [edit snmp v3]

Release Information

Statement introduced before Junos OS Release 7.4.
type inform option added in Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Select management targets for SNMPv3 notifications as well as the type of notifications. Notifications can be either traps or informs.

Options

name—Name assigned to the notification.

tag-name—Notifications are sent to all targets configured with this tag.

type—Notification type is **trap** or **inform**. Traps are unconfirmed notifications. Informs are confirmed notifications.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Inform Notification Type and Target Address on page 142
- Configuring the SNMPv3 Trap Notification on page 131

notify-filter (Applying to the Management Target)

Syntax notify-filter *profile-name*;

Hierarchy Level [edit snmp v3 target-parameters *target-parameters-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the notify filter applied to a specific set of SNMPv3 target parameters. Target parameters are the message processing and security parameters for notifications sent to a target SNMP manager.

Options ***profile-name***—Name of the notify filter to apply to notifications.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Applying the Trap Notification Filter on page 137

notify-filter (Configuring the Profile Name)

Syntax
```
notify-filter profile-name {
  oid oid (include | exclude);
}
```

Hierarchy Level [edit snmp v3]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Specify a group of MIB objects for which you define access. The notify filter limits the type of traps or informs sent to the network management system.

Options ***profile-name***—Name assigned to the notify filter.

The remaining statement is explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Trap Notification Filter on page 132
- oid (SNMP) on page 231

notify-view

Syntax notify-view *view-name*;

Hierarchy Level [edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Associate the notify view with a community (for SNMPv1 or SNMPv2c clients) or a group name (for SNMPv3 clients).

Options *view-name*—Name of the view to which the SNMP user group has access.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring MIB Views on page 109
- Configuring the Notify View on page 125

oid (SNMP)

Syntax oid *oid* (include | exclude);

Hierarchy Level [edit snmp v3 notify-filter *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify an object identifier (OID) used to represent a subtree of MIB objects. This OID is a prefix that the represented MIB objects have in common.

Options **exclude**—Exclude the subtree of MIB objects represented by the specified OID.

include—Include the subtree of MIB objects represented by the specified OID.

oid—Object identifier used to represent a subtree of MIB objects. All MIB objects represented by this statement have the specified OID as a prefix. You can specify the OID using either a sequence of dotted integers or a subtree name.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Trap Notification Filter on page 132

parameters

Syntax

```
parameters {
  message-processing-model (v1 | v2c | v3);
  security-level (none | authentication | privacy);
  security-model (usm | v1 | v2c);
  security-name security-name;
}
```

Hierarchy Level [edit snmp v3 target-parameters *target-parameters-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure a set of target parameters for message processing and security.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Defining and Configuring the Trap Target Parameters on page 136

port (SNMP)

Syntax port *port-number*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure a UDP port number for an SNMP target.

Default If you omit this statement, the default port is 162.

Options ***port-number***—Port number for the SNMP target.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Port on page 134

privacy-3des

Syntax

```
privacy-3des {
    privacy-password privacy-password;
}
```

Hierarchy Level

[edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description

Configure the triple Data Encryption Standard (3DES) as the privacy type for the SNMPv3 user.

Options

privacy-password *privacy-password*—Password that a user enters. The password is then converted into a key that is used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Encryption Type on page 120

privacy-aes128

Syntax

```
privacy-aes128 {
    privacy-password privacy-password;
}
```

Hierarchy Level

[edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description

Configure the Advanced Encryption Standard encryption algorithm (CFB128-AES-128 Privacy Protocol) for the SNMPv3 user.

Options

privacy-password *privacy-password*—Password that a user enters. The password is then converted into a key that is used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Encryption Type on page 120

privacy-des

Syntax

```
privacy-des {
    privacy-password privacy-password;
}
```

Hierarchy Level [edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the Data Encryption Standard (DES) as the privacy type for the SNMPv3 user.

Options **privacy-password *privacy-password***—Password that a user enters. The password is then converted into a key that is used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Encryption Type on page 120

privacy-none

Syntax

```
privacy-none;
```

Hierarchy Level [edit snmp v3 usm local-engine user *username*],
[edit snmp v3 usm remote-engine *engine-id* user *username*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure that no encryption be used for the SNMPv3 user.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Encryption Type on page 120

privacy-password

Syntax privacy-password *privacy-password*;

Hierarchy Level [edit snmp v3 usm local-engine user *username* privacy-3des],
[edit snmp v3 usm local-engine user *username* privacy-aes128],
[edit snmp v3 usm local-engine user *username* privacy-des],
[edit snmp v3 usm remote-engine *engine-id* user *username* privacy-3des],
[edit snmp v3 usm remote-engine *engine-id* user *username* privacy-aes128],
[edit snmp v3 usm remote-engine *engine-id* user *username* privacy-des]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure a privacy password for the SNMPv3 user.

Options *privacy-password*—Password that a user enters. The password is then converted into a key that is used for encryption.

SNMPv3 has special requirements when you create plain-text passwords on a router or switch:

- The password must be at least eight characters long.
- The password can include alphabetic, numeric, and special characters, but it cannot include control characters.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Encryption Type on page 120

read-view

Syntax read-view *view-name*;

Hierarchy Level [edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Associate the read-only view with a community (for SNMPv1 or SNMPv2c clients) or a group name (for SNMPv3 clients).

Options ***view-name***—The name of the view to which the SNMP user group has access.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Read View on page 126
- Configuring MIB Views on page 109

remote-engine

Syntax

```
remote-engine engine-id {
  user username {
    authentication-md5 {
      authentication-password authentication-password;
    }
    authentication-none;
    authentication-sha {
      authentication-password authentication-password;
    }
    privacy-aes128 {
      privacy-password privacy-password;
    }
    privacy-des {
      privacy-password privacy-password;
    }
    privacy-3des {
      privacy-password privacy-password;
    }
    privacy-none {
      privacy-password privacy-password;
    }
  }
}
```

Hierarchy Level [edit snmp v3 usm]

Release Information Statement introduced in Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the remote engine information for the user-based security model (USM). To send inform messages to an SNMPv3 user on a remote device, you must configure the engine identifier for the SNMP agent on the remote device where the user resides.

Options ***engine-id***—Specify engine identifier in hexadecimal format. Used to compute the security digest for authenticating and encrypting packets sent to a user on the remote host.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Remote Engine and Remote User on page 140

routing-instance (SNMPv3)

Syntax routing-instance *routing-instance-name*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced in Junos OS Release 8.3.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify a routing instance for an SNMPv3 trap target.

Options ***routing-instance-name***—Name of the routing instance.

To configure a routing instance within a logical system, specify the logical system name followed by the routing instance name. Use a slash (/) to separate the two names (for example, **test-ls/test-ri**). To configure the default routing instance on a logical system, specify the logical system name followed by **default** (for example, **test-ls/default**).

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Trap Target Address on page 133

security-level (Defining Access Privileges)

Syntax

```
security-level (authentication | none | privacy) {
  notify-view view-name;
  read-view view-name;
  write-view view-name;
}
```

Hierarchy Level

[edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c)]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description

Define the security level used for access privileges.

Default

none

Options

authentication—Provide authentication but no encryption.

none—No authentication and no encryption.

privacy—Provide authentication and encryption.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Level on page 124

security-level (Generating SNMP Notifications)

Syntax security-level (authentication | none | privacy);

Hierarchy Level [edit snmp v3 target-parameters *target-parameters-name* parameters]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the security level to use when generating SNMP notifications.

Default none

Options **authentication**—Provide authentication but no encryption.

none—No authentication and no encryption.

privacy—Provide authentication and encryption.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Level on page 138

security-model (Access Privileges)

Syntax security-model (usm | v1 | v2c);

Hierarchy Level [edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the security model for an SNMPv3 group. The security model is used to determine access privileges for the group.

Options **usm**—SNMPv3 security model.

v1—SNMPv1 security model.

v2c—SNMPv2c security model.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Model on page 124

security-model (Group)

Syntax

```
security-model (usm | v1 | v2c) {
  security-name security-name {
    group group-name;
  }
}
```

Hierarchy Level [edit snmp v3 vacm security-to-group]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Define a security model for an SNMPv3 group and associate the security name of a user with the group. All users in the group have the same access privileges.

Options

usm—SNMPv3 security model.

v1—SNMPv1 security model.

v2c—SNMPv2c security model.

Required Privilege Level
snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Model on page 128

security-model (SNMP Notifications)

Syntax security-model (usm | v1 | v2c);

Hierarchy Level [edit snmp v3 target-parameters *target-parameters-name* parameters]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the security model for an SNMPv3 group. The security model is used for SNMP notifications.

Options

usm—SNMPv3 security model.

v1—SNMPv1 security model.

v2c—SNMPv2c security model.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Model on page 138

security-name (Community String)

Syntax security-name *security-name*;

Hierarchy Level [edit snmp v3 snmp-community *community-index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Associate a community string with the security name of a user. The community string, which is used for SNMPv1 and SNMPv2c clients in an SNMPv3 system, is configured at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level.

Options ***security-name***—Name that is used for messaging security and user access control.

NOTE: The security name must match the configured security name at the **[edit snmp v3 target-parameters *target-parameters-name* parameters]** hierarchy level when you configure traps or informs.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Names on page 145

security-name (Security Group)

Syntax

```
security-name security-name {
    group group-name;
}
```

Hierarchy Level [edit snmp v3 vacm security-to-group security-model (usm | v1 | v2c)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Associate the security name of a user (for SNMPv3 clients) or a community string (for SNMPv1 and SNMPv2c clients) with a configured security group.

Options *security-name*—SNMPv3 secure username configured at the **[edit snmp v3 usm local-engine user *username*]** hierarchy level that is used for messaging security. For SNMPv1 and SNMPv2c, the security name is the community string configured at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Assigning Security Names to Groups on page 128
- *Assigning a Security Name to a Group*

security-name (SNMP Notifications)

Syntax security-name *security-name*;

Hierarchy Level [edit snmp v3 target-parameters *target-parameters-name* parameters]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the security name used when generating SNMP notifications.

Options ***security-name***—If the SNMPv3 USM security model is used, identify the user when generating the SNMP notification. If the v1 or v2c security models are used, identify the SNMP community used when generating the notification.

NOTE: The access privileges for the group associated with this security name must allow this notification to be sent.

If you are using the v1 or v2 security models, the security name at the **[edit snmp v3 vacm security-to-group]** hierarchy level must match the security name at the **[edit snmp v3 snmp-community *community-index*]** hierarchy level.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Security Name on page 139

security-to-group

Syntax

```
security-to-group {
  security-model (usm | v1 | v2c) {
    group group-name;
    security-name security-name;
  }
}
```

Hierarchy Level [edit snmp v3 vacm]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the group to which a specific SNMPv3 security name belongs. The security name is used for messaging security.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Assigning Security Model and Security Name to a Group on page 127

snmp-community

Syntax

```
snmp-community community-index {
  community-name community-name;
  security-name security-name;
  tag tag-name;
}
```

Hierarchy Level [edit snmp v3]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the SNMP community which authorizes SNMPv1 or SNMPv2c clients in an SNMPv3 system.

Options ***community-index***—(Optional) String that identifies an SNMP community.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the SNMPv3 Community on page 144

tag (SNMPv3)

Syntax tag *tag-name*;

Hierarchy Level [edit snmp v3 notify *name*],
[edit snmp v3 snmp-community *community-index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure a set of targets to receive traps or informs (for IPv4 packets only).

Options ***tag-name***—Identifies the address of managers that are allowed to use a community string.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Tag on page 146
- Configuring the SNMPv3 Trap Notification on page 131

tag-list

Syntax tag-list *tag-list*;

Hierarchy Level [edit snmp v3 target-address *target-address-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure an SNMP tag list used to select target addresses.

Options ***tag-list***—Define sets of target addresses (tags). To specify more than one tag, specify the tag names as a space-separated list enclosed within double quotes.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Trap Target Address on page 134

target-address

Syntax

```
target-address target-address-name {
  address address;
  address-mask address-mask;
  logical-system (SNMP logical-system;
  port (SNMP) port-number;
  retry-count (SNMPv3) number;
  routing-instance (SNMPv3) instance;
  tag-list tag-list;
  target-parameters target-parameters-name;
  timeout seconds;
}
```

Hierarchy Level [edit snmp v3]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure the address of an SNMP management application and the parameters to be used in sending notifications.

Options *target-address-name*—String that identifies the target address.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Trap Target Address on page 133

target-parameters

Syntax

At the **[edit snmp v3]** hierarchy level:

```
target-parameters target-parameters-name {
  profile-name;
  parameters {
    message-processing-model (v1 | v2c | V3);
    security-level (authentication | none | privacy);
    security-model (usm | v1 | v2c);
    security-name security-name;
  }
}
```

At the **[edit snmp v3 target-address *target-address-name*]** hierarchy level:

```
target-parameters target-parameters-name;
```

Hierarchy Level

[edit snmp v3]
[edit snmp v3 target-address *target-address-name*]

Release Information

Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description

Configure the message processing and security parameters for sending notifications to a particular management target. The target parameters are configured at the **[edit snmp v3]** hierarchy level. The remaining statements at this level are explained separately.

Then apply the target parameters configured at the **[edit snmp v3 target-parameters *target-parameters-name*]** hierarchy level to the target address configuration at the **[edit snmp v3]** hierarchy level.

Required Privilege Level

snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Defining and Configuring the Trap Target Parameters on page 136
- Applying Target Parameters on page 135

type (SNMPv3)

Syntax type (inform | trap);

Hierarchy Level [edit snmp v3 notify *name*]

Release Information Statement introduced before Junos OS Release 7.4.
inform option added in Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure the type of SNMP notification.

Options **inform**—Defines the type of notification as an inform. SNMP informs are confirmed notifications.

trap—Defines the type of notification as a trap. SNMP traps are unconfirmed notifications.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring SNMP Informs on page 139
- Configuring the SNMPv3 Trap Notification on page 131

user

Syntax user *username*;

Hierarchy Level [edit snmp v3 usm local-engine],
[edit snmp v3 usm remote-engine *engine-id*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Specify a user associated with an SNMPv3 group on a local or remote SNMP engine.

Options *username*—SNMPv3 user-based security model (USM) username.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Creating SNMPv3 Users on page 118

usm

Syntax

```
usm {
  local-engine {
    user username {
      authentication-md5 {
        authentication-password authentication-password;
      }
      authentication-none;
      authentication-sha {
        authentication-password authentication-password;
      }
      privacy-aes128 {
        privacy-password privacy-password;
      }
      privacy-des {
        privacy-password privacy-password;
      }
      privacy-3des {
        privacy-password privacy-password;
      }
      privacy-none {
        privacy-password privacy-password;
      }
    }
    remote-engine engine-id {
      user username {
        authentication-md5 {
          authentication-password authentication-password;
        }
        authentication-none;
        authentication-sha {
          authentication-password authentication-password;
        }
        privacy-aes128 {
          privacy-password privacy-password;
        }
        privacy-des {
          privacy-password privacy-password;
        }
        privacy-3des {
          privacy-password privacy-password;
        }
        privacy-none {
          privacy-password privacy-password;
        }
      }
    }
  }
}
```

Hierarchy Level [edit snmp v3]

Release Information Statement introduced before Junos OS Release 7.4.

Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure user-based security model (USM) information.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Creating SNMPv3 Users on page 118
- Configuring the Remote Engine and Remote User on page 140

v3

Syntax

```
v3 {
  notify name {
    tag tag-name;
    type trap;
  }
  notify-filter profile-name {
    oid object-identifier (include | exclude);
  }
  snmp-community community-index {
    community-name community-name;
    security-name security-name;
    tag tag-name;
  }
  target-address target-address-name {
    address address;
    address-mask address-mask;
    logical-system (SNMP logical-system;
    port port-number;
    retry-count number;
    routing-instance instance;
    tag-list tag-list;
    target-parameters target-parameters-name;
    timeout seconds;
  }
  target-parameters target-parameters-name {
    notify-filter profile-name;
    parameters {
      message-processing-model (v1 | v2c | V3);
      security-level (authentication | none | privacy);
      security-model (usm | v1 | v2c);
      security-name security-name;
    }
  }
  usm {
    local-engine {
      user username {
        authentication-md5 {
          authentication-password authentication-password;
        }
        authentication-sha {
          authentication-password authentication-password;
        }
        authentication-none;
        privacy-aes128 {
          privacy-password privacy-password;
        }
        privacy-des {
          privacy-password privacy-password;
        }
        privacy-des {
          privacy-password privacy-password;
        }
```

```
            privacy-none;
          }
        }
        remote-engine engine-id {
          user username {
            authentication-md5 {
              authentication-password authentication-password;
            }
            authentication-sha {
              authentication-password authentication-password;
            }
            authentication-none;
            privacy-aes128 {
              privacy-password privacy-password;
            }
            privacy-des {
              privacy-password privacy-password;
            }
            privacy-3des {
              privacy-password privacy-password;
            }
            privacy-none {
              privacy-password privacy-password;
            }
          }
        }
      }
      vacm {
        access {
          group group-name {
            (default-context-prefix | context-prefix context-prefix){
              security-model (any | usm | v1 | v2c) {
                security-level (authentication | none | privacy) {
                  notify-view view-name;
                  read-view view-name;
                  write-view view-name;
                }
              }
            }
          }
        }
        security-to-group {
          security-model (usm | v1 | v2c) {
            security-name security-name {
              group group-name;
            }
          }
        }
      }
    }
```

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure SNMPv3.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Minimum SNMPv3 Configuration on a Device Running Junos OS on page 116

vacm

Syntax

```
vacm {
  access {
    group group-name {
      (default-context-prefix | context-prefix context-prefix){
        security-model (any | usm | v1 | v2c) {
          security-level (authentication | none | privacy) {
            notify-view view-name;
            read-view view-name;
            write-view view-name;
          }
        }
      }
    }
  }
  security-to-group {
    security-model (usm | v1 | v2c);
    security-name security-name {
      group group-name;
    }
  }
}
```

Hierarchy Level [edit snmp v3]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure view-based access control model (VACM) information, including access privileges such as security model and security level for a group of users.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Defining Access Privileges for an SNMP Group on page 122

write-view

Syntax write-view *view-name*;

Hierarchy Level [edit snmp v3 vacm access group *group-name* (default-context-prefix | context-prefix *context-prefix*) security-model (any | usm | v1 | v2c) security-level (authentication | none | privacy)]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series switches.

Description Associate the write view with a community (for SNMPv1 or SNMPv2c clients) or a group name (for SNMPv3 clients).

Options ***view-name***—Name of the view for which the SNMP user group has write permission.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring MIB Views on page 109
- Configuring the Write View on page 126

CHAPTER 13

RMON Configuration Statements

alarm (SNMP RMON)

Syntax

```
alarm index {
  description description;
  falling-event-index index;
  falling-threshold integer;
  falling-threshold-interval seconds;
  interval seconds;
  request-type (get-next-request | get-request | walk-request);
  rising-event-index index;
  rising-threshold integer;
  sample-type (absolute-value | delta-value);
  startup-alarm (falling-alarm | rising-alarm | rising-or-falling alarm);
  syslog-subtag syslog-subtag;
  variable oid-variable;
}
```

Hierarchy Level [edit snmp rmon]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.
Statement introduced in Junos OS Release 11.1 for the QFX Series.

Description Configure RMON alarm entries.

Options *index*—Identifies this alarm entry as an integer.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring an Alarm Entry and Its Attributes on page 158
- event (SNMP) on page 264
- *Configuring RMON Alarms and Events*
- *RMON MIB Event, Alarm, Log, and History Control Tables*
- *Monitoring RMON MIB Tables*
- *Understanding RMON*
- *Junos OS Network Management Configuration Guide*

community (SNMP RMON)

Syntax community *community-name*;

Hierarchy Level [edit snmp rmon event *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The trap group that is used when generating a trap (if **eventType** is configured to send traps). If that trap group has the **rmon-alarm** trap category configured, a trap is sent to all the targets configured for that trap group. The community string in the trap matches the name of the trap group (and hence, the value of **eventCommunity**). If nothing is configured, traps are sent to each group with the **rmon-alarm** category set.

Options ***community-name***—Identifies the trap group that is used when generating a trap if the event is configured to send traps.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring an Event Entry and Its Attributes on page 162

description (SNMP RMON)

Syntax description *description*;

Hierarchy Level [edit snmp rmon alarm *index*],
[edit snmp rmon event *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Text description of alarm or event.

Options ***description***—Text description of an alarm or event entry. If the description includes spaces, enclose it in quotation marks (" ").

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Description on page 159
- Configuring an Event Entry and Its Attributes on page 162

event (SNMP)

Syntax

```
event index {
  community community-name;
  description description;
  type type;
}
```

Hierarchy Level [edit snmp rmon]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure RMON event entries.

Options *index*—Identifier for a specific event entry.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring an Event Entry and Its Attributes on page 162
- alarm on page 262

falling-event-index

Syntax falling-event-index *index*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The index of the event entry that is used when a falling threshold is crossed. If this value is zero, no event is triggered.

Options *index*—Index of the event entry that is used when a falling threshold is crossed.

Range: 0 through 65,535

Default: 0

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Falling Event Index or Rising Event Index on page 159
- rising-event-index on page 268

falling-threshold

Syntax falling-threshold *integer*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The lower threshold for the sampled variable. When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval is greater than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is less than or equal to this threshold, and the associated startup-alarm value is equal to falling-alarm value or rising-or-falling-alarm value. After a falling event is generated, another falling event cannot be generated until the sampled value rises above this threshold and reaches the **rising-threshold**.

Options ***integer***—The lower threshold for the alarm entry.

Range: -2,147,483,648 through 2,147,483,647

Default: 20 percent less than **rising-threshold**

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Falling Threshold or Rising Threshold on page 159
- rising-threshold (SNMP RMON) on page 268

falling-threshold-interval

Syntax falling-threshold-interval *seconds*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced in Junos OS Release 8.3.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Interval between samples when the rising threshold is crossed. Once the alarm crosses the falling threshold, the regular sampling interval is used.

Options ***seconds***—Time between samples, in seconds.
Range: 1 through 2,147,483,647 seconds
Default: 60 seconds

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Falling Threshold Interval on page 160
- interval (SNMP RMON) on page 266

interval (SNMP RMON)

Syntax interval *seconds*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Interval between samples.

Options ***seconds***—Time between samples, in seconds.
Range: 1 through 2,147,483,647 seconds
Default: 60 seconds

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Interval on page 160

request-type

Syntax request-type (get-next-request | get-request | walk-request);

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced in Junos OS Release 8.3.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Extend monitoring to a specific SNMP object instance (**get-request**), or extend monitoring to all object instances belonging to a MIB branch (**walk-request**), or extend monitoring to the next object instance after the instance specified in the configuration (**get-next-request**).

Options **get-next-request**—Performs an SNMP get next request.

get-request—Performs an SNMP get request.

walk-request—Performs an SNMP walk request.
Default: walk-request

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Request Type on page 160
- variable on page 271

rising-event-index

Syntax rising-event-index *index*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Index of the event entry that is used when a rising threshold is crossed. If this value is zero, no event is triggered.

Options ***index***—Index of the event entry that is used when a rising threshold is crossed.
Range: 0 through 65,535
Default: 0

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Falling Event Index or Rising Event Index on page 159
- falling-event-index on page 264

rising-threshold (SNMP RMON)

Syntax rising-threshold *integer*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Upper threshold for the sampled variable. When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval is less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is greater than or equal to this threshold, and the associated startup alarm value is equal to the falling alarm or rising or falling alarm value. After a rising event is generated, another rising event cannot be generated until the sampled value falls below this threshold and reaches the falling threshold.

Options ***integer***—The lower threshold for the alarm entry.
Range: –2,147,483,648 through 2,147,483,647

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Falling Threshold or Rising Threshold on page 159
- falling-threshold on page 265

rmon

Syntax rmon { ... }

Hierarchy Level [edit snmp]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure Remote Monitoring.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring an Alarm Entry and Its Attributes on page 158

sample-type

Syntax sample-type (absolute-value | delta-value);

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Method of sampling the selected variable.

Options
absolute-value—Actual value of the selected variable is used when comparing against the thresholds.

delta-value—Difference between samples of the selected variable is used when comparing against the thresholds.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Sample Type on page 161

startup-alarm

Syntax startup-alarm (falling-alarm | rising-alarm | rising-or-falling-alarm);

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The alarm that can be sent upon entry startup.

Options

falling-alarm—Generated if the first sample after the alarm entry becomes active is less than or equal to the falling threshold.

rising-alarm—Generated if the first sample after the alarm entry becomes active is greater than or equal to the rising threshold.

rising-or-falling-alarm—Generated if the first sample after the alarm entry becomes active satisfies either of the corresponding thresholds.

Default: rising-or-falling-alarm

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Startup Alarm on page 161

syslog-subtag

Syntax syslog-subtag *syslog-subtag*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced in Junos OS Release 8.5.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Add a tag to the system log message.

Options

syslog-subtag *syslog-subtag*—Tag of not more than 80 uppercase characters to be added to syslog messages.

Default: None

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the System Log Tag on page 161

type (SNMP RMON)

Syntax type *type*;

Hierarchy Level [edit snmp rmon event *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Type of notification generated when a threshold is crossed.

Options *type*—Type of notification:

- **log**—Add an entry to **logTable**.
- **log-and-trap**—Send an SNMP trap and make a log entry.
- **none**—No notifications are sent.
- **snmptrap**—Send an SNMP trap.

Default: log-and-trap

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring an Event Entry and Its Attributes on page 162

variable

Syntax variable *oid-variable*;

Hierarchy Level [edit snmp rmon alarm *index*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Object identifier (OID) of MIB variable to be monitored.

Options *oid-variable*—OID of the MIB variable that is being monitored. The OID can be a dotted decimal (for example, **1.3.6.1.2.1.2.1.2.2.1.10.1**). Alternatively, use the MIB object name (for example, **ifInOctets.1**).

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuring the Variable on page 162

CHAPTER 14

Health Monitoring Configuration Statements

falling-threshold

Syntax falling-threshold *percentage*;

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The lower threshold is expressed as a percentage of the maximum possible value for the sampled variable. When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval is greater than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is less than or equal to this threshold. After a falling event is generated, another falling event cannot be generated until the sampled value rises above this threshold and reaches the **rising-threshold**.

Options *percentage*—The lower threshold for the alarm entry.

Range: 1 through 100

Default: 70 percent of the maximum possible value

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Falling Threshold or Rising Threshold on page 165
- rising-threshold (SNMP Health Monitor) on page 275

health-monitor

Syntax

```
health-monitor {
  falling-threshold percentage;
  interval seconds;
  rising-threshold percentage;
}
```

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure health monitoring.

The remaining statements are explained separately.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring Health Monitoring on Devices Running Junos OS on page 163

interval (SNMP Health Monitor)

Syntax interval *seconds*;

Hierarchy Level [edit snmp health-monitor]

Release Information Statement introduced in Junos OS Release 8.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Interval between samples.

Options ***seconds***—Time between samples, in seconds.

Range: 1 through 2147483647 seconds

Default: 300 seconds

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring the Interval on page 166

rising-threshold (SNMP Health Monitor)

Syntax rising-threshold *percentage*;

Hierarchy Level [edit snmp]

Release Information Statement introduced in Junos OS Release 8.0.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description The upper threshold is expressed as a percentage of the maximum possible value for the sampled variable. When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval is less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is greater than or equal to this threshold. After a rising event is generated, another rising event cannot be generated until the sampled value falls below this threshold and reaches the **falling-threshold**.

Options *percentage*—The lower threshold for the alarm entry.

Range: 1 through 100

Default: 80 percent of the maximum possible value

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- falling-threshold on page 273
- Configuring the Falling Threshold or Rising Threshold on page 165

CHAPTER 15

Accounting Options Configuration Statements

accounting-options

Syntax
```
accounting-options {...}
}
```

Hierarchy Level [edit]

Release Information Statement introduced before Junos OS Release 7.4.

Description Configure options for accounting statistics collection.

Required Privilege Level
snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation
- Configuration Statements at the [edit accounting-options] Hierarchy Level on page 167
- Accounting Options Configuration on page 168

archive-sites

Syntax

```
archive-sites {
  site-name;
}
```

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Configure an archive site. If more than one site name is configured, an ordered list of archive sites for the accounting-data log files is created. When a file is archived, the router or switch attempts to transfer the file to the first URL in the list, moving to the next site only if the transfer does not succeed. The log file is stored at the archive site with a filename of the format ***router-name_log-filename_timestamp***.

Options ***site-name***—Any valid FTP URL to a destination. For information about specifying valid FTP URLs, see the *Junos System Basics: Getting Started Configuration Guide*.

Required Privilege Level snmp—To view this statement in the configuration.
snmp-control—To add this statement to the configuration.

Related Documentation

- Configuring Archive Sites on page 174

class-usage-profile

Syntax

```
class-usage-profile profile-name {
  file filename;
  interval minutes;
  source-classes {
    source-class-name;
  }
  destination-classes {
    destination-class-name;
  }
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a class usage profile, which is used to log class usage statistics to a file in the **/var/log** directory. The class usage profile logs class usage statistics for the configured source classes on every interface that has **destination-class-usage** configured.

For information about configuring source classes, see the *Junos Routing Protocols Configuration Guide*. For information about configuring source class usage, see the *Junos Network Management Configuration Guide*.

Options ***profile-name***—Name of the destination class profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Class Usage Profiles on page 185

counters

Syntax

```
counters {
  counter-name;
}
```

Hierarchy Level [edit accounting-options filter-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Names of counters for which filter profile statistics are collected. The packet and byte counts for the counters are logged to a file in the **/var/log** directory.

Options ***counter-name***—Name of the counter.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Counters on page 178

destination-classes

Syntax

```
destination-classes {
  destination-class-name;
}
```

Hierarchy Level [edit accounting-options class-usage-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the destination classes for which statistics are collected.

Options ***destination-class-name***—Name of the destination class to include in the source class usage profile.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring a Class Usage Profile on page 185

fields (for Interface Profiles)

Syntax

```
fields {
  field-name;
}
```

Hierarchy Level [edit accounting-options interface-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Statistics to collect in an accounting-data log file for an interface.

Options *field-name*—Name of the field:

- **input-bytes**—Input bytes
- **input-errors**—Generic input error packets
- **input-multicast**—Input packets arriving by multicast
- **input-packets**—Input packets
- **input-unicast**—Input unicast packets
- **output-bytes**—Output bytes
- **output-errors**—Generic output error packets
- **output-multicast**—Output packets sent by multicast
- **output-packets**—Output packets
- **output-unicast**—Output unicast packets

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Interface Profile on page 175

fields (for Routing Engine Profiles)

Syntax

```
fields {
  field-name;
}
```

Hierarchy Level [edit accounting-options routing-engine-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.

Description Statistics to collect in an accounting-data log file for a Routing Engine.

Options ***field-name***—Name of the field:

- **cpu-load-1**—Average system load over the last 1 minute
- **cpu-load-5**—Average system load over the last 5 minutes
- **cpu-load-15**—Average system load over the last 15 minutes
- **date**—Date, in YYYYMMDD format
- **host-name**—Hostname for the router
- **time-of-day**—Time of day, in HHMMSS format
- **uptime**—Time since last reboot, in seconds

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Routing Engine Profile on page 189

file (Associating with a Profile)

Syntax file *filename*;

Hierarchy Level [edit accounting-options class-usage-profile *profile-name*],
[edit accounting-options filter-profile *profile-name*],
[edit accounting-options interface-profile *profile-name*],
[edit accounting-options mib-profile *profile-name*],
[edit accounting-options routing-engine-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
The **[edit accounting-options mib-profile *profile-name*]** hierarchy added in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series Switches.

Description Specify the accounting log file associated with the profile.

Options ***filename***—Name of the log file. You must specify a filename already configured in the **file** statement at the **[edit accounting-options]** hierarchy level.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Interface Profile on page 175
- Configuring the Filter Profile on page 177
- Configuring the MIB Profile on page 187
- Configuring the Routing Engine Profile on page 189

file (Configuring a Log File)

Syntax

```
file filename {
  archive-sites {
    site-name;
  }
  files number;
  nonpersistent;
  size bytes;
  source-classes time;
  transfer-interval minutes;
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify a log file to be used for accounting data.

Options ***filename***—Name of the file in which to write accounting data.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring Accounting-Data Log Files on page 172

files

Syntax files *number*;

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the maximum number of log files to be used for accounting data.

Options ***number***—The maximum number of files. When a log file (for example, **profilelog**) reaches its maximum size, it is renamed **profilelog.0**, then **profilelog.1**, and so on, until the maximum number of log files is reached. Then the oldest log file is overwritten. The minimum value for ***number*** is 3 and the default value is 10.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring Accounting-Data Log Files on page 172

filter-profile

Syntax

```
filter-profile profile-name {
  counters {
    counter-name;
  }
  file filename;
  interval minutes;
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a profile to filter and collect packet and byte count statistics and write them to a file in the **/var/log** directory. To apply the profile to a firewall filter, you include the **accounting-profile** statement at the **[edit firewall filter *filter-name*]** hierarchy level. For more information about firewall filters, see the *Junos Network Management Configuration Guide*.

Options ***profile-name***—Name of the filter profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Filter Profile on page 177

interface-profile

Syntax

```
interface-profile profile-name {
  fields {
    field-name;
  }
  file filename;
  interval minutes;
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a profile to filter and collect error and packet statistics and write them to a file in the **/var/log** directory. You can specify an interface profile for either a physical or a logical interface.

Options ***profile-name***—Name of the interface profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the Interface Profile on page 175

interval (Accounting Options)

Syntax interval *minutes*;

Hierarchy Level
[edit accounting-options class-usage-profile *profile-name*],
[edit accounting-options filter-profile *profile-name*],
[edit accounting-options interface-profile *profile-name*],
[edit accounting-options mib-profile *profile-name*],
[edit accounting-options routing-engine-profile *profile-name*]

Release Information
Statement introduced before Junos OS Release 7.4.
The **[edit accounting-options mib-profile *profile-name*]** hierarchy level added in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify how often statistics are collected for the accounting profile.

Options
minutes—Length of time between each collection of statistics.
Range: 1 through 2880 minutes
Default: 30 minutes

Required Privilege Level
interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Interface Profile on page 175
- Configuring the Filter Profile on page 177
- Configuring the MIB Profile on page 187
- Configuring the Routing Engine Profile on page 189

mib-profile

Syntax

```
mib-profile profile-name {
  file filename;
  interval minutes;
  object-names {
    mib-object-name;
  }
  operation operation-name;
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a MIB profile to collect selected MIB statistics and write them to a file in the **/var/log** directory.

NOTE: Do not configure MIB objects related to interface octets or packets for a MIB profile, because it can cause the SNMP walk or a CLI show command to time out.

Options *profile-name*—Name of the MIB statistics profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation

- Configuring the MIB Profile on page 187

nonpersistent

Syntax nonpersistent;

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced in Junos OS Release 8.3.

Description For J Series Services Routers only. Store log files used for accounting data in the **mfs/var/log** directory (located on DRAM) instead of the **cf/var/log** directory (located on the compact flash drive). This feature is useful for minimizing read/write traffic on the router's compact flash drive.

NOTE: If log files for accounting data are stored on DRAM, these files are lost when you reboot the router. Therefore, you should back up these files periodically.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Storage Location of the File on page 172

object-names

Syntax
```
object-names {
  mib-object-name;
}
```

Hierarchy Level [edit accounting-options mib-profile *profile-name*]

Release Information Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the name of each MIB object for which MIB statistics are collected for an accounting-data log file.

Options ***mib-object-name***—Name of a MIB object. You can specify more than one MIB object name.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the MIB Profile on page 187

operation

Syntax operation *operation-name*;

Hierarchy Level [edit accounting-options mib-profile *profile-name*]

Release Information Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the name of the operation used to collect MIB statistics for an accounting-data log file.

Options ***operation-name***—Name of the operation to use. You can specify a **get**, **get-next**, or **walk** operation.

Default: walk

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the MIB Profile on page 187

routing-engine-profile

Syntax
```
routing-engine-profile profile-name {
  fields {
    field-name;
  }
  file filename;
  interval minutes;
}
```

Hierarchy Level [edit accounting-options]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Create a Routing Engine profile to collect selected Routing Engine statistics and write them to a file in the **/var/log** directory.

Options ***profile-name***—Name of the Routing Engine statistics profile.

The remaining statements are explained separately.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Routing Engine Profile on page 189

size

Syntax size *bytes*;

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify attributes of an accounting-data log file.

Options ***bytes***—Maximum size of each log file, in bytes, kilobytes (KB), megabytes (MB), or gigabytes (GB). When a log file (for example, **profilelog**) reaches its maximum size, it is renamed **profilelog.0**, then **profilelog.1**, and so on, until the maximum number of log files is reached. Then the oldest log file is overwritten. If you do not specify a size, the file is closed, archived, and renamed when the time specified for the transfer interval is exceeded.

Syntax: ***x*** to specify bytes, ***x***k to specify KB, ***x***m to specify MB, ***x***g to specify GB

Range: 256 KB through 1 GB

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Maximum Size of the File on page 173

source-classes

Syntax
```
source-classes {
  source-class-name;
}
```

Hierarchy Level [edit accounting-options class-usage-profile *profile-name*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the source classes for which statistics are collected.

Options ***source-class-name***—Name of the source class to include in the class usage profile.

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring a Class Usage Profile on page 185

start-time (Log File Transfer)

Syntax start-time *time*;

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced in Junos OS Release 8.2.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the start time for transfer of an accounting-data log file.

Options *time*—Start time for file transfer.

Syntax: *YYYY-MM-DD.hh:mm*

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Start Time for File Transfer on page 173

transfer-interval

Syntax transfer-interval *minutes*;

Hierarchy Level [edit accounting-options file *filename*]

Release Information Statement introduced before Junos OS Release 7.4.
Statement introduced in Junos OS Release 9.0 for EX Series switches.

Description Specify the length of time the file remains open and receives new statistics before it is closed and transferred to an archive site.

Options *minutes*—Time the file remains open and receives new statistics before it is closed and transferred to an archive site.

Range: 5 through 2880 minutes

Default: 30 minutes

Required Privilege Level interface—To view this statement in the configuration.
interface-control—To add this statement to the configuration.

Related Documentation
- Configuring the Transfer Interval of the File on page 173

PART 3

Administration

CHAPTER 16

SNMP

Loading MIB Files to a Network Management System

For your network management system (NMS) to identify and understand the MIB objects used by the Junos OS, you must first load the MIB files to your NMS using a MIB compiler. A MIB compiler is a utility that parses the MIB information such as the MIB object name, IDs, and data type for the NMS.

You can download the Junos MIB package from the **Enterprise-Specific MIBs and Traps** section of the Junos OS Technical Publications index page at http://www.juniper.net/techpubs/software/junos/index.html . The Junos MIB package is available in **.zip** and **.tar** packages. You can download the appropriate format based on your requirements.

The Junos MIB package contains two folders: **StandardMibs** and **JuniperMibs**. The **StandardMibs** folder contains the standard MIBs and RFCs that are supported on devices running the Junos OS, whereas the **JuniperMibs** folder contains the Juniper Networks enterprise-specific MIBs.

To load MIB files that are required for managing and monitoring devices running the Junos OS:

1. Go to the Junos OS Technical Publications index page (http://www.juniper.net/techpubs/software/junos/index.html).
2. Click the tab that corresponds to the Junos OS Release for which you want to download the MIB files.
3. On the selected tab, click the **+** (plus) sign that corresponds to the **Enterprise-Specific MIBs and Traps** section to expand the section.
4. Click the **TAR** or **ZIP** link that corresponds to the **Enterprise MIBs** link under the **Enterprise-Specific MIBs and Traps** section to download the Junos MIB package.
5. Decompress the file (**.tar** or **.zip**) using an appropriate utility.
6. Load the standard MIB files (from the **StandardMibs** folder) in the following order:

NOTE: Some of the MIB compilers that are commonly used have the standard MIBs preloaded on them. If the standard MIBs are already loaded on the MIB compiler that you are using, skip this step and proceed to Step 7.

a. **mib-SNMPv2-SMI.txt**
b. **mib-SNMPv2-TC.txt**
c. **mib-IANAifType-MIB.txt**
d. **mib-IANA-RTPROTO-MIB.txt**
e. **mib-rfc1907.txt**
f. **mib-rfc2011a.txt**
g. **mib-rfc2012a.txt**
h. **mib-rfc2013a.txt**
i. **mib-rfc2863a.txt**

7. Load the remaining standard MIB files.

NOTE: You must follow the order specified in this procedure, and ensure that all standard MIBs are loaded before you load the enterprise-specific MIBs. There might be dependencies that require a particular MIB to be present on the compiler before loading some other MIB. You can find such dependencies listed in the **IMPORT** section of the MIB file.

8. Load the Juniper Networks enterprise-specific SMI MIB, **mib-jnx-smi.txt**, and the following optional SMI MIBs based on your requirements:
   - **mib-jnx-js-smi.txt**—(Optional) For Juniper Security MIB tree objects
   - **mib-jnx-ex-smi.txt**—(Optional) For EX Series Ethernet Switches
   - **mib-jnx-exp.txt**—(Recommended) For Juniper Networks experimental MIB objects
9. Load the remaining enterprise-specific MIBs from the **JuniperMibs** folder.

TIP: While loading a MIB file, if the compiler returns an error message saying that any of the objects is undefined, open the MIB file using a text editor and ensure that all the MIB files listed in the **IMPORT** section are loaded on the compiler. If any of the MIB files listed in the **IMPORT** section is not loaded on the compiler, load that MIB file, and then try to load the MIB file that failed to load.

For example, the enterprise-specific PING MIB, **mib-jnx-ping.txt**, has dependencies on RFC 2925, DiSMAN-PING-MIB, **mib-rfc2925a.txt**. If you try to load **mib-jnx-ping.txt** before loading **mib-rfc2925a.txt**, the compiler returns

an error message saying that certain objects in mib-jnx-ping.txt are undefined. Load mib-rfc2925a.txt, and then try to load mib-jnx-ping.txt. The enterprise-specific PING MIB, mib-jnx-ping.txt, then loads without any issue.

Related Documentation

- Standard SNMP MIBs Supported by Junos OS on page 19
- Juniper Networks Enterprise-Specific MIBs on page 35

Tracing SNMP Activity on a Device Running Junos OS

SNMP tracing operations track activity for SNMP agents and record the information in log files. The logged error descriptions provide detailed information to help you solve problems faster.

By default, Junos OS does not trace any SNMP activity. If you include the **traceoptions** statement at the **[edit snmp]** hierarchy level, the default tracing behavior is:

- Important activities are logged in files located in the **/var/log** directory. Each log is named after the SNMP agent that generates it. Currently, the following log files are created in the **/var/log** directory when the **traceoptions** statement is used:
  - chassisd
  - craftd
  - ilmid
  - mib2d
  - rmopd
  - serviced
  - snmpd
- When a trace file named ***filename*** reaches its maximum size, it is renamed ***filename*.0**, then ***filename*.1**, and so on, until the maximum number of trace files is reached. Then the oldest trace file is overwritten. (For more information about how log files are created, see the *Junos OS System Log Messages Reference*.)
- Log files can be accessed only by the user who configured the tracing operation.

You cannot change the directory (**/var/log**) in which trace files are located. However, you can customize the other trace file settings by including the following statements at the **[edit snmp]** hierarchy level:

```
[edit snmp]
traceoptions {
   file <files number> <match regular-expression> <size size> <world-readable |
      no-world-readable>;
   flag flag;
   no-remote-trace;
}
```

These statements are described in the following sections:

Configuring the Number and Size of SNMP Log Files

By default, when the trace file reaches 128 kilobytes (KB) in size, it is renamed ***filename*.0**, then ***filename*.1**, and so on, until there are three trace files. Then the oldest trace file (***filename*.2**) is overwritten.

You can configure the limits on the number and size of trace files by including the following statements at the **[edit snmp traceoptions]** hierarchy level:

```
[edit snmp traceoptions]
file files number size size;
```

For example, set the maximum file size to 2 MB, and the maximum number of files to 20. When the file that receives the output of the tracing operation (***filename***) reaches 2 MB, ***filename*** is renamed ***filename*.0**, and a new file called ***filename*** is created. When the new ***filename*** reaches 2 MB, ***filename*.0** is renamed ***filename*.1** and ***filename*** is renamed ***filename*.0**. This process repeats until there are 20 trace files. Then the oldest file (***filename*.19**) is overwritten by the newest file (***filename*.0**).

The number of files can be from 2 through 1000 files. The file size of each file can be from 10 KB through 1 gigabyte (GB).

Configuring Access to the Log File

By default, log files can be accessed only by the user who configured the tracing operation.

To specify that any user can read all log files, include the **file world-readable** statement at the **[edit snmp traceoptions]** hierarchy level:

```
[edit snmp traceoptions]
file world-readable;
```

To explicitly set the default behavior, include the **file no-world-readable** statement at the **[edit snmp traceoptions]** hierarchy level:

```
[edit snmp traceoptions]
file no-world-readable;
```

Configuring a Regular Expression for Lines to Be Logged

By default, the trace operation output includes all lines relevant to the logged activities.

You can refine the output by including the **match** statement at the **[edit snmp traceoptions file *filename*]** hierarchy level and specifying a regular expression (regex) to be matched:

```
[edit snmp traceoptions]
file filename match regular-expression;
```

Configuring the Trace Operations

By default, only important activities are logged. You can specify which trace operations are to be logged by including the following **flag** statement (with one or more tracing flags) at the **[edit snmp traceoptions]** hierarchy level:

```
[edit snmp traceoptions]
flag {
   all;
   configuration;
   database;
   events;
   general;
   interface-stats;
   nonvolatile-sets;
   pdu;
   policy;
   protocol-timeouts;
   routing-socket;
   server;
   subagent;
   timer;
   varbind-error;
}
```

Table 13 on page 301 describes the meaning of the SNMP tracing flags.

Table 13: SNMP Tracing Flags

Flag	Description	Default Setting
all	Log all operations.	Off
configuration	Log reading of the configuration at the **[edit snmp]** hierarchy level.	Off
database	Log events involving storage and retrieval in the events database.	Off
events	Log important events.	Off
general	Log general events.	Off
interface-stats	Log physical and logical interface statistics.	Off
nonvolatile-set	Log nonvolatile SNMP set request handling.	Off
pdu	Log SNMP request and response packets.	Off
policy	Log policy processing.	Off
protocol-timeouts	Log SNMP response timeouts.	Off
routing-socket	Log routing socket calls.	Off

Table 13: SNMP Tracing Flags *(continued)*

Flag	Description	Default Setting
server	Log communication with processes that are generating events.	Off
subagent	Log subagent restarts.	Off
timer	Log internal timer events.	Off
varbind-error	Log variable binding errors.	Off

To display the end of the log for an agent, issue the **show log *agent*d | last** operational mode command:

```
[edit]
user@host# run show log agentd | last
```

where ***agent*** is the name of an SNMP agent.

Related Documentation

- Configuring SNMP on a Device Running Junos OS on page 91
- Configuration Statements at the [edit snmp] Hierarchy Level on page 87
- Example: Tracing SNMP Activity on page 111
- *Configuring SNMP*

CHAPTER 17

Remote Monitoring, Health Monitoring, and Service Quality Monitoring

Starting a Ping Test

Before you start a ping test, configure a Ping MIB view. This allows SNMP **Set** requests on **pingMIB**. To start a ping test, create a row in **pingCtlTable** and set **pingCtlAdminStatus** to **enabled**. The minimum information that must be specified before setting **pingCtlAdminStatus** to **enabled** is:

- **pingCtlOwnerIndexSnmpAdminString**
- **pingCtlTestNameSnmpAdminString**

- **pingCtlTargetAddressInetAddress**
- **pingCtlTargetAddressTypeInetAddressType**
- **pingCtlRowStatusRowStatus**

For all other values, defaults are chosen unless otherwise specified. **pingCtlOwnerIndex** and **pingCtlTestName** are used as the index, so their values are specified as part of the object identifier (OID). To create a row, set **pingCtlRowStatus** to **createAndWait** or **createAndGo** on a row that does not already exist. A value of **active** for **pingCtlRowStatus** indicates that all necessary information has been supplied and the test can begin; **pingCtlAdminStatus** can be set to **enabled**. An SNMP **Set** request that sets **pingCtlRowStatus** to **active** will fail if the necessary information in the row is not specified or is inconsistent. For information about how to configure a view, see "Setting SNMP Views" on page 14.

There are two ways to start a ping test:

- Using Multiple Set Protocol Data Units (PDUs) on page 304
- Using a Single Set PDU on page 304

Using Multiple Set Protocol Data Units (PDUs)

You can use multiple **Set** request PDUs (multiple PDUs, with one or more varbinds each) and set the following variables in this order to start the test:

- **pingCtlRowStatus** to **createAndWait**
- All appropriate test variables
- **pingCtlRowStatus** to **active**

  Junos OS now verifies that all necessary information to run a test has been specified.
- **pingCtlAdminStatus** to **enabled**

Using a Single Set PDU

You can use a single **Set** request PDU (one PDU, with multiple varbinds) to set the following variables to start the test:

- **pingCtlRowStatus** to **createAndGo**
- All appropriate test variables
- **pingCtlAdminStatus** to **enabled**

Monitoring a Running Ping Test

When **pingCtlAdminStatus** is successfully set to **enabled**, the following is done before the acknowledgment of the SNMP **Set** request is sent back to the client:

- **pingResultsEntry** is created if it does not already exist.
- **pingResultsOperStatus** transitions to **enabled**.

For more information, see the following sections:

- pingResultsTable on page 305
- pingProbeHistoryTable on page 306
- Generating Traps on page 307

pingResultsTable

While the test is running, **pingResultsEntry** keeps track of the status of the test. The value of **pingResultsOperStatus** is **enabled** while the test is running and **disabled** when it has stopped.

The value of **pingCtlAdminStatus** remains **enabled** until you set it to **disabled**. Thus, to get the status of the test, you must examine **pingResultsOperStatus**.

The **pingCtlFrequency** variable can be used to schedule many tests for one **pingCtlEntry**. After a test ends normally (you did not stop the test) and the **pingCtlFrequency** number of seconds has elapsed, the test is started again just as if you had set **pingCtlAdminStatus** to **enabled**. If you intervene at any time between repeated tests (you set **pingCtlAdminStatus** to **disabled** or **pingCtlRowStatus** to **notInService**), the repeat feature is disabled until another test is started and ends normally. A value of 0 for **pingCtlFrequency** indicates this repeat feature is not active.

pingResultsIpTgtAddr and **pingResultsIpTgtAddrType** are set to the value of the resolved destination address when the value of **pingCtlTargetAddressType** is **dns**. When a test starts successfully and **pingResultsOperStatus** transitions to **enabled**:

- **pingResultsIpTgtAddr** is set to **null-string**.
- **pingResultsIpTgtAddrType** is set to **unknown**.

pingResultsIpTgtAddr and **pingResultsIpTgtAddrType** are not set until **pingCtlTargetAddress** can be resolved to a numeric address. To retrieve these values, poll **pingResultsIpTgtAddrType** for any value other than **unknown** after successfully setting **pingCtlAdminStatus** to **enabled**.

At the start of a test, **pingResultsSentProbes** is initialized to 1 and the first probe is sent. **pingResultsSentProbes** increases by 1 each time a probe is sent.

As the test runs, every **pingCtlTimeOut** seconds, the following occur:

- **pingProbeHistoryStatus** for the corresponding **pingProbeHistoryEntry** in **pingProbeHistoryTable** is set to **requestTimedOut**.
- A **pingProbeFailed** trap is generated, if necessary.
- An attempt is made to send the next probe.

NOTE: No more than one outstanding probe exists for each test.

For every probe, you can receive one of the following results:

- The target host acknowledges the probe with a response.
- The probe times out; there is no response from the target host acknowledging the probe.
- The probe could not be sent.

Each probe result is recorded in **pingProbeHistoryTable**. For more information about **pingProbeHistoryTable**, see "pingProbeHistoryTable" on page 306.

When a response is received from the target host acknowledging the current probe:

- **pingResultsProbeResponses** increases by 1.
- The following variables are updated:
  - **pingResultsMinRtt**—Minimum round-trip time
  - **pingResultsMaxRtt**—Maximum round-trip time
  - **pingResultsAverageRtt**—Average round-trip time
  - **pingResultsRttSumOfSquares**—Sum of squares of round-trip times
  - **pingResultsLastGoodProbe**—Timestamp of the last response

NOTE: Only probes that result in a response from the target host contribute to the calculation of the round-trip time (RTT) variables.

When a response to the last probe is received or the last probe has timed out, the test is complete.

pingProbeHistoryTable

An entry in **pingProbeHistoryTable** (**pingProbeHistoryEntry**) represents a probe result and is indexed by three variables:

- The first two variables, **pingCtlOwnerIndex** and **pingCtlTestName**, are the same ones used for **pingCtlTable**, which identifies the test.
- The third variable, **pingProbeHistoryIndex**, is a counter to uniquely identify each probe result.

The maximum number of **pingProbeHistoryTable** entries created for a given test is limited by **pingCtlMaxRows**. If **pingCtlMaxRows** is set to 0, no **pingProbeHistoryTable** entries are created for that test.

Each time a probe result is determined, a **pingProbeHistoryEntry** is created and added to **pingProbeHistoryTable**. **pingProbeHistoryIndex** of the new **pingProbeHistoryEntry** is 1 greater than the last **pingProbeHistoryEntry** added to **pingProbeHistoryTable** for that test. **pingProbeHistoryIndex** is set to 1 if this is the first entry in the table. The same test can be run multiple times, so this index keeps growing.

If **pingProbeHistoryIndex** of the last **pingProbeHistoryEntry** added is 0xFFFFFFFF, the next **pingProbeHistoryEntry** added has **pingProbeHistoryIndex** set to 1.

The following are recorded for each probe result:

- **pingProbeHistoryResponse**—Time to live (TTL)
- **pingProbeHistoryStatus**—What happened and why
- **pingProbeHistoryLastRC**—Return code (RC) value of ICMP packet
- **pingProbeHistoryTime**—Timestamp when probe result was determined

When a probe cannot be sent, **pingProbeHistoryResponse** is set to 0. When a probe times out, **pingProbeHistoryResponse** is set to the difference between the time when the probe was discovered to be timed out and the time when the probe was sent.

Generating Traps

For any trap to be generated, the appropriate bit of **pingCtlTrapGeneration** must be set. You must also configure a trap group to receive remote operations. A trap is generated under the following conditions:

- A **pingProbeFailed** trap is generated every time **pingCtlTrapProbeFailureFilter** number of consecutive probes fail during the test.
- A **pingTestFailed** trap is generated when the test completes and at least **pingCtlTrapTestFailureFilter** number of probes fail.
- A **pingTestCompleted** trap is generated when the test completes and fewer than **pingCtlTrapTestFailureFilter** probes fail.

NOTE: **A probe is considered a failure when pingProbeHistoryStatus of the probe result is anything besides responseReceived.**

For information about how to configure a trap group to receive remote operations, see "Configuring SNMP Trap Groups" on page 104 and "Example: Setting Trap Notification for Remote Operations" on page 15.

Gathering Ping Test Results

You can either poll **pingResultsOperStatus** to find out when the test is complete or request that a trap be sent when the test is complete. For more information about **pingResultsOperStatus**, see "pingResultsTable" on page 305. For more information about Ping MIB traps, see "Generating Traps" on page 307.

The statistics calculated and then stored in **pingResultsTable** include:

- **pingResultsMinRtt**—Minimum round-trip time
- **pingResultsMaxRtt**—Maximum round-trip time
- **pingResultsAverageRtt**—Average round-trip time

- **pingResultsProbeResponses**—Number of responses received
- **pingResultsSentProbes**—Number of attempts to send probes
- **pingResultsRttSumOfSquares**—Sum of squares of round-trip times
- **pingResultsLastGoodProbe**—Timestamp of the last response

You can also consult **pingProbeHistoryTable** for more detailed information about each probe. The index used for **pingProbeHistoryTable** starts at 1, goes to 0xFFFFFFFF, and wraps to 1 again.

For example, if **pingCtlProbeCount** is 15 and **pingCtlMaxRows** is 5, then upon completion of the first run of this test, **pingProbeHistoryTable** contains probes like those in Table 14 on page 308.

Table 14: Results in pingProbeHistoryTable: After the First Ping Test

pingProbeHistoryIndex	Probe Result
11	Result of 11th probe from run 1
12	Result of 12th probe from run 1
13	Result of 13th probe from run 1
14	Result of 14th probe from run 1
15	Result of 15th probe from run 1

Upon completion of the first probe of the second run of this test, **pingProbeHistoryTable** will contain probes like those in Table 15 on page 308.

Table 15: Results in pingProbeHistoryTable: After the First Probe of the Second Test

pingProbeHistoryIndex	Probe Result
12	Result of 12th probe from run 1
13	Result of 13th probe from run 1
14	Result of 14th probe from run 1
15	Result of 15th probe from run 1
16	Result of 1st probe from run 2

Upon completion of the second run of this test, **pingProbeHistoryTable** will contain probes like those in Table 16 on page 309.

Table 16: Results in pingProbeHistoryTable: After the Second Ping Test

pingProbeHistoryIndex	Probe Result
26	Result of 11th probe from run 2
27	Result of 12th probe from run 2
28	Result of 13th probe from run 2
29	Result of 14th probe from run 2
30	Result of 15th probe from run 2

History entries can be deleted from the MIB in two ways:

- More history entries for a given test are added and the number of history entries exceeds **pingCtlMaxRows**. The oldest history entries are deleted to make room for the new ones.
- You delete the entire test by setting **pingCtlRowStatus** to **destroy**.

Stopping a Ping Test

To stop an active test, set **pingCtlAdminStatus** to **disabled**. To stop the test and remove its **pingCtlEntry**, **pingResultsEntry**, and any **pingHistoryEntry** objects from the MIB, set **pingCtlRowStatus** to **destroy**.

Interpreting Ping Variables

This section clarifies the ranges for the following variables that are not explicitly specified in the Ping MIB:

- **pingCtlDataSize**—The value of this variable represents the total size of the payload (in bytes) of an outgoing probe packet. This payload includes the timestamp (8 bytes) that is used to time the probe. This is consistent with the definition of **pingCtlDataSize** (maximum value of 65,507) and the standard ping application.

  If the value of **pingCtlDataSize** is between 0 and 8 inclusive, it is ignored and the payload is 8 bytes (the timestamp). The Ping MIB assumes all probes are timed, so the payload must always include the timestamp.

  For example, if you wish to add an additional 4 bytes of payload to the packet, you must set **pingCtlDataSize** to 12.

- **pingCtlDataFill**—The first 8 bytes of the data segment of the packet is for the timestamp. After that, the **pingCtlDataFill** pattern is used in repetition. The default pattern (when **pingCtlDataFill** is not specified) is (00, 01, 02, 03 ... FF, 00, 01, 02, 03 ... FF, ...).
- **pingCtlMaxRows**—The maximum value is 255.

- **pingMaxConcurrentRequests**—The maximum value is 500.
- **pingCtlTrapProbeFailureFilter** and **pingCtlTrapTestFailureFilter**—A value of 0 for **pingCtlTrapProbeFailureFilter** or **pingCtlTrapTestFailureFilter** is not well defined by the Ping MIB. If **pingCtlTrapProbeFailureFilter** is 0, **pingProbeFailed** traps will not be generated for the test under any circumstances. If **pingCtlTrapTestFailureFilter** is 0, **pingTestFailed** traps will not be generated for the test under any circumstances.

Starting a Traceroute Test

Before you start a traceroute test, configure a Traceroute MIB view. This allows SNMP **Set** requests on **tracerouteMIB**. To start a test, create a row in **traceRouteCtlTable** and set **traceRouteCtlAdminStatus** to **enabled**. You must specify at least the following before setting **traceRouteCtlAdminStatus** to **enabled**:

- **traceRouteCtlOwnerIndexSnmpAdminString**
- **traceRouteCtlTestNameSnmpAdminString**
- **traceRouteCtlTargetAddressInetAddress**
- **traceRouteCtlRowStatusRowStatus**

For all other values, defaults are chosen unless otherwise specified. **traceRouteCtlOwnerIndex** and **traceRouteCtlTestName** are used as the index, so their values are specified as part of the OID. To create a row, set **traceRouteCtlRowStatus** to **createAndWait** or **createAndGo** on a row that does not already exist. A value of **active** for **traceRouteCtlRowStatus** indicates that all necessary information has been specified and the test can begin; **traceRouteCtlAdminStatus** can be set to **enabled**. An SNMP **Set** request that sets **traceRouteCtlRowStatus** to **active** will fail if the necessary information in the row is not specified or is inconsistent. For information about how to configure a view, see "Setting SNMP Views" on page 14.

There are two ways to start a traceroute test:

- Using Multiple Set PDUs on page 310
- Using a Single Set PDU on page 311

Using Multiple Set PDUs

You can use multiple **Set** request PDUs (multiple PDUs, with one or more varbinds each) and set the following variables in this order to start the test:

- **traceRouteCtlRowStatus** to createAndWait
- All appropriate test variables
- **traceRouteCtlRowStatus** to **active**

  The Junos OS now verifies that all necessary information to run a test has been specified.
- **traceRouteCtlAdminStatus** to **enabled**

Using a Single Set PDU

You can use a single **Set** request PDU (one PDU, with multiple varbinds) to set the following variables to start the test:

- **traceRouteCtlRowStatus** to **createAndGo**
- All appropriate test variables
- **traceRouteCtlAdminStatus** to enabled

Related Documentation

- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77
- Monitoring a Running Traceroute Test on page 311
- SNMP Remote Operations Overview on page 13
- Monitoring Traceroute Test Completion on page 315
- Gathering Traceroute Test Results on page 316
- Stopping a Traceroute Test on page 317
- Interpreting Traceroute Variables on page 318

Monitoring a Running Traceroute Test

When **traceRouteCtlAdminStatus** is successfully set to **enabled**, the following is done before the acknowledgment of the SNMP **Set** request is sent back to the client:

- **traceRouteResultsEntry** is created if it does not already exist.
- **traceRouteResultsOperStatus** transitions to enabled.

For more information, see the following sections:

- traceRouteResultsTable on page 311
- traceRouteProbeResultsTable on page 312
- traceRouteHopsTable on page 313
- Generating Traps on page 315

traceRouteResultsTable

While the test is running, this **traceRouteResultsTable** keeps track of the status of the test. The value of **traceRouteResultsOperStatus** is **enabled** while the test is running and **disabled** when it has stopped.

The value of **traceRouteCtlAdminStatus** remains **enabled** until you set it to **disabled**. Thus, to get the status of the test, you must examine **traceRouteResultsOperStatus**.

The **traceRouteCtlFrequency** variable can be used to schedule many tests for one **traceRouteCtlEntry**. After a test ends normally (you did not stop the test) and **traceRouteCtlFrequency** number of seconds has elapsed, the test is started again just as

if you had set **traceRouteCtlAdminStatus** to **enabled**. If you intervene at any time between repeated tests (you set **traceRouteCtlAdminStatus** to **disabled** or **traceRouteCtlRowStatus** to **notInService**), the repeat feature is **disabled** until another test is started and ends normally. A value of 0 for **traceRouteCtlFrequency** indicates this repeat feature is not active.

traceRouteResultsIpTgtAddr and **traceRouteResultsIpTgtAddrType** are set to the value of the resolved destination address when the value of **traceRouteCtlTargetAddressType** is **dns**. When a test starts successfully and **traceRouteResultsOperStatus** transitions to **enabled**:

- **traceRouteResultsIpTgtAddr** is set to null-string.
- **traceRouteResultsIpTgtAddrType** is set to unknown.

traceRouteResultsIpTgtAddr and **traceRouteResultsIpTgtAddrType** are not set until **traceRouteCtlTargetAddress** can be resolved to a numeric address. To retrieve these values, poll **traceRouteResultsIpTgtAddrType** for any value other than **unknown** after successfully setting **traceRouteCtlAdminStatus** to **enabled**.

At the start of a test, **traceRouteResultsCurHopCount** is initialized to **traceRouteCtlInitialTtl**, and **traceRouteResultsCurProbeCount** is initialized to 1. Each time a probe result is determined, **traceRouteResultsCurProbeCount** increases by 1. While the test is running, the value of **traceRouteResultsCurProbeCount** reflects the current outstanding probe for which results have not yet been determined.

The **traceRouteCtlProbesPerHop** number of probes is sent for each time-to-live (TTL) value. When the result of the last probe for the current hop is determined, provided that the current hop is not the destination hop, **traceRouteResultsCurHopCount** increases by 1, and **traceRouteResultsCurProbeCount** resets to 1.

At the start of a test, if this is the first time this test has been run for this **traceRouteCtlEntry**, **traceRouteResultsTestAttempts** and **traceRouteResultsTestSuccesses** are initialized to 0.

At the end of each test execution, **traceRouteResultsOperStatus** transitions to **disabled**, and **traceRouteResultsTestAttempts** increases by 1. If the test was successful in determining the full path to the target, **traceRouteResultsTestSuccesses** increases by 1, and **traceRouteResultsLastGoodPath** is set to the current time.

traceRouteProbeResultsTable

Each entry in **traceRouteProbeHistoryTable** is indexed by five variables:

- The first two variables, **traceRouteCtlOwnerIndex** and **traceRouteCtlTestName**, are the same ones used for **traceRouteCtlTable** and to identify the test.
- The third variable, **traceRouteProbeHistoryIndex**, is a counter, starting from 1 and wrapping at FFFFFFFF. The maximum number of entries is limited by **traceRouteCtlMaxRows**.
- The fourth variable, **traceRouteProbeHistoryHopIndex**, indicates which hop this probe is for (the actual time-to-live or TTL value). Thus, the first **traceRouteCtlProbesPerHop**

number of entries created when a test starts have a value of **traceRouteCtlInitialTtl** for **traceRouteProbeHistoryHopIndex**.

- The fifth variable, **traceRouteProbeHistoryProbeIndex**, is the probe for the current hop. It ranges from 1 to **traceRouteCtlProbesPerHop**.

While a test is running, as soon as a probe result is determined, the next probe is sent. A maximum of **traceRouteCtlTimeOut** seconds elapses before a probe is marked with status **requestTimedOut** and the next probe is sent. There is never more than one outstanding probe per traceroute test. Any probe result coming back after a probe times out is ignored.

Each probe can:

- Result in a response from a host acknowledging the probe
- Time out with no response from a host acknowledging the probe
- Fail to be sent

Each probe status is recorded in **traceRouteProbeHistoryTable** with **traceRouteProbeHistoryStatus** set accordingly.

Probes that result in a response from a host record the following data:

- **traceRouteProbeHistoryResponse**—Round-trip time (RTT)
- **traceRouteProbeHistoryHAddrType**—The type of HAddr (next argument)
- **traceRouteProbeHistoryHAddr**—The address of the hop

All probes, regardless of whether a response for the probe is received, have the following recorded:

- **traceRouteProbeHistoryStatus**—What happened and why
- **traceRouteProbeHistoryLastRC**—Return code (RC) value of the ICMP packet
- **traceRouteProbeHistoryTime**—Timestamp when the probe result was determined

When a probe cannot be sent, **traceRouteProbeHistoryResponse** is set to 0. When a probe times out, **traceRouteProbeHistoryResponse** is set to the difference between the time when the probe was discovered to be timed out and the time when the probe was sent.

traceRouteHopsTable

Entries in **traceRouteHopsTable** are indexed by three variables:

- The first two, **traceRouteCtlOwnerIndex** and **traceRouteCtlTestName**, are the same ones used for **traceRouteCtlTable** and identify the test.
- The third variable, **traceRouteHopsHopIndex**, indicates the current hop, which starts at 1 (not **traceRouteCtlInitialTtl**).

When a test starts, all entries in **traceRouteHopsTable** with the given **traceRouteCtlOwnerIndex** and **traceRouteCtlTestName** are deleted. Entries in this table are only created if **traceRouteCtlCreateHopsEntries** is set to **true**.

A new **traceRouteHopsEntry** is created each time the first probe result for a given TTL is determined. The new entry is created whether or not the first probe reaches a host. The value of **traceRouteHopsHopIndex** is increased by 1 for this new entry.

NOTE: Any **traceRouteHopsEntry** can lack a value for **traceRouteHopsIpTgtAddress** if there are no responses to the probes with the given TTL.

Each time a probe reaches a host, the IP address of that host is available in the probe result. If the value of **traceRouteHopsIpTgtAddress** of the current **traceRouteHopsEntry** is not set, then the value of **traceRouteHopsIpTgtAddress** is set to this IP address. If the value of **traceRouteHopsIpTgtAddress** of the current **traceRouteHopsEntry** is the same as the IP address, then the value does not change. If the value of **traceRouteHopsIpTgtAddress** of the current **traceRouteHopsEntry** is different from this IP address, indicating a path change, a new **traceRouteHopsEntry** is created with:

- **traceRouteHopsHopIndex** variable increased by 1
- **traceRouteHopsIpTgtAddress** set to the IP address

NOTE: A new entry for a test is added to **traceRouteHopsTable** each time a new TTL value is used or the path changes. Thus, the number of entries for a test may exceed the number of different TTL values used.

When a probe result is determined, the value **traceRouteHopsSentProbes** of the current **traceRouteHopsEntry** increases by 1. When a probe result is determined, and the probe reaches a host:

- The value **traceRouteHopsProbeResponses** of the current **traceRouteHopsEntry** is increased by 1.
- The following variables are updated:
  - **traceRouteResultsMinRtt**—Minimum round-trip time
  - **traceRouteResultsMaxRtt**—Maximum round-trip time
  - **traceRouteResultsAverageRtt**—Average round-trip time
  - **traceRouteResultsRttSumOfSquares**—Sum of squares of round-trip times
  - **traceRouteResultsLastGoodProbe**—Timestamp of the last response

NOTE: Only probes that reach a host affect the round-trip time values.

Generating Traps

For any trap to be generated, the appropriate bit of **traceRouteCtlTrapGeneration** must be set. You must also configure a trap group to receive remote operations. Traps are generated under the following conditions:

- **traceRouteHopsIpTgtAddress** of the current probe is different from the last probe with the same TTL value (**traceRoutePathChange**).
- A path to the target could not be determined (**traceRouteTestFailed**).

A path to the target was determined (**traceRouteTestCompleted**).

For information about how to configure a trap group to receive remote operations, see "Configuring SNMP Trap Groups" on page 104 and "Example: Setting Trap Notification for Remote Operations" on page 15.

Monitoring Traceroute Test Completion

When a test is complete, **traceRouteResultsOperStatus** transitions from **enabled** to **disabled**. This transition occurs in the following situations:

- The test ends successfully. A probe result indicates that the destination has been reached. In this case, the current hop is the last hop. The rest of the probes for this hop are sent. When the last probe result for the current hop is determined, the test ends.
- **traceRouteCtlMaxTtl** threshold is exceeded. The destination is never reached. The test ends after the number of probes with TTL value equal to **traceRouteCtlMaxttl** have been sent.
- **traceRouteCtlMaxFailures** threshold is exceeded. The number of consecutive probes that end with status **requestTimedOut** exceeds **traceRouteCtlMaxFailures**.
- You end the test. You set **traceRouteCtlAdminStatus** to **disabled** or delete the row by setting **traceRouteCtlRowStatus** to **destroy**.
- You misconfigured the traceroute test. A value or variable you specified in **traceRouteCtlTable** is incorrect and will not allow a single probe to be sent. Because of the nature of the data, this error could not be determined until the test was started; that is, until after **traceRouteResultsOperStatus** transitioned to **enabled**. When this occurs, one entry is added to **traceRouteProbeHistoryTable** with **traceRouteProbeHistoryStatus** set to the appropriate error code.

If **traceRouteCtlTrapGeneration** is set properly, either the **traceRouteTestFailed** or **traceRouteTestCompleted** trap is generated.

Related Documentation

- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77
- Monitoring a Running Traceroute Test on page 311
- SNMP Remote Operations Overview on page 13
- Starting a Traceroute Test on page 310

Gathering Traceroute Test Results

You can either poll **traceRouteResultsOperStatus** to find out when the test is complete or request that a trap be sent when the test is complete. For more information about **traceResultsOperStatus**, see "traceRouteResultsTable" on page 311. For more information about Traceroute MIB traps, see the Generating Traps section in "Monitoring a Running Traceroute Test" on page 311.

Statistics are calculated on a per-hop basis and then stored in **traceRouteHopsTable**. They include the following for each hop:

- **traceRouteHopsIpTgtAddressType**—Address type of host at this hop
- **traceRouteHopsIpTgtAddress**—Address of host at this hop
- **traceRouteHopsMinRtt**—Minimum round-trip time
- **traceRouteHopsMaxRtt**—Maximum round-trip time
- **traceRouteHopsAverageRtt**—Average round-trip time
- **traceRouteHopsRttSumOfSquares**—Sum of squares of round-trip times
- **traceRouteHopsSentProbes**—Number of attempts to send probes
- **traceRouteHopsProbeResponses**—Number of responses received
- **traceRouteHopsLastGoodProbe**—Timestamp of last response

You can also consult **traceRouteProbeHistoryTable** for more detailed information about each probe. The index used for **traceRouteProbeHistoryTable** starts at 1, goes to 0xFFFFFFFF, and wraps to 1 again.

For example, assume the following:

- **traceRouteCtlMaxRows** is 10.
- **traceRouteCtlProbesPerHop** is 5.
- There are eight hops to the target (the target being number eight).
- Each probe sent results in a response from a host (the number of probes sent is not limited by **traceRouteCtlMaxFailures**).

In this test, 40 probes are sent. At the end of the test, **traceRouteProbeHistoryTable** would have a history of probes like those in Table 17 on page 317.

Table 17: traceRouteProbeHistoryTable

HistoryIndex	HistoryHopIndex	HistoryProbeIndex
31	7	1
32	7	2
33	7	3
34	7	4
35	7	5
36	8	1
37	8	2
38	8	3
39	8	4
40	8	5

Related Documentation

- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77
- Monitoring a Running Traceroute Test on page 311
- SNMP Remote Operations Overview on page 13
- Starting a Traceroute Test on page 310
- Monitoring Traceroute Test Completion on page 315
- Stopping a Traceroute Test on page 317
- Interpreting Traceroute Variables on page 318

Stopping a Traceroute Test

To stop an active test, set **traceRouteCtlAdminStatus** to **disabled**. To stop a test and remove its **traceRouteCtlEntry**, **traceRouteResultsEntry**, **traceRouteProbeHistoryEntry**, and **traceRouteProbeHistoryEntry** objects from the MIB, set **traceRouteCtlRowStatus** to **destroy**.

Related Documentation

- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77
- Monitoring a Running Traceroute Test on page 311
- SNMP Remote Operations Overview on page 13
- Starting a Traceroute Test on page 310
- Monitoring Traceroute Test Completion on page 315

- Gathering Traceroute Test Results on page 316
- Interpreting Traceroute Variables on page 318

Interpreting Traceroute Variables

This topic contains information about the ranges for the following variables that are not explicitly specified in the Traceroute MIB:

- **traceRouteCtlMaxRows**—The maximum value for **traceRouteCtlMaxRows** is 2550. This represents the maximum TTL (255) multiplied by the maximum for **traceRouteCtlProbesPerHop** (10). Therefore, the **traceRouteProbeHistoryTable** accommodates one complete test at the maximum values for one **traceRouteCtlEntry**. Usually, the maximum values are not used and the **traceRouteProbeHistoryTable** is able to accommodate the complete history for many tests for the same **traceRouteCtlEntry**.
- **traceRouteMaxConcurrentRequests**—The maximum value is 50. If a test is running, it has one outstanding probe. **traceRouteMaxConcurrentRequests** represents the maximum number of traceroute tests that have **traceRouteResultsOperStatus** with a value of **enabled**. Any attempt to start a test with **traceRouteMaxConcurrentRequests** tests running will result in the creation of one probe with **traceRouteProbeHistoryStatus** set to **maxConcurrentLimitReached** and that test will end immediately.
- **traceRouteCtlTable**—The maximum number of entries allowed in this table is 100. Any attempt to create a 101st entry will result in a **BAD_VALUE** message for SNMPv1 and a **RESOURCE_UNAVAILABLE** message for SNMPv2.

Related Documentation

- Using the Traceroute MIB for Remote Monitoring Devices Running Junos OS on page 77
- Monitoring a Running Traceroute Test on page 311
- SNMP Remote Operations Overview on page 13
- Starting a Traceroute Test on page 310
- Monitoring Traceroute Test Completion on page 315
- Gathering Traceroute Test Results on page 316
- Stopping a Traceroute Test on page 317

Using alarmTable to Monitor MIB Objects

To use **alarmTable** to monitor a MIB object, perform the following tasks:

- Creating an Alarm Entry on page 319
- Configuring the Alarm MIB Objects on page 319
- Activating a New Row in alarmTable on page 321
- Modifying an Active Row in alarmTable on page 321
- Deactivating a Row in alarmTable on page 322

Creating an Alarm Entry

To create an alarm entry, first create a new row in **alarmTable** using the **alarmStatus** object. For example, create alarm #1 using the UCD command-line utilities:

```
snmpset -Os -v2c router community alarmStatus.1 i createRequest
```

Configuring the Alarm MIB Objects

Once you have created the new row in **alarmTable**, configure the following Alarm MIB objects:

NOTE: Other than **alarmStatus**, you cannot modify any of the objects in the entry if the associated **alarmStatus** object is set to valid.

- alarmInterval on page 319
- alarmVariable on page 319
- alarmSampleType on page 320
- alarmValue on page 320
- alarmStartupAlarm on page 320
- alarmRisingThreshold on page 320
- alarmFallingThreshold on page 320
- alarmOwner on page 321
- alarmRisingEventIndex on page 321
- alarmFallingEventIndex on page 321

alarmInterval

The interval, in seconds, over which data is sampled and compared with the rising and falling thresholds. For example, to set **alarmInterval** for alarm #1 to 30 seconds, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmInterval.1 i 30
```

alarmVariable

The object identifier of the variable to be sampled. During a **Set** request, if the supplied variable name is not available in the selected MIB view, a **badValue** error is returned. If at any time the variable name of an established **alarmEntry** is no longer available in the selected MIB view, the probe changes the status of **alarmVariable** to invalid. For example, to identify **ifInOctets.61** as the variable to be monitored, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmVariable.1 o .1.3.6.1.2.1.2.2.1.10.61
```

alarmSampleType

The method of sampling the selected variable and calculating the value to be compared against the thresholds. If the value of this object is **absoluteValue**, the value of the selected variable is compared directly with the thresholds at the end of the sampling interval. If the value of this object is **deltaValue**, the value of the selected variable at the last sample is subtracted from the current value, and the difference is compared with the thresholds. For example, to set **alarmSampleType** for alarm #1 to **deltaValue**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmSampleType.1 i deltaValue
```

alarmValue

The value of the variable during the last sampling period. This value is compared with the rising and falling thresholds. If the sample type is **deltaValue**, this value equals the difference between the samples at the beginning and end of the period. If the sample type is **absoluteValue**, this value equals the sampled value at the end of the period.

alarmStartupAlarm

An alarm that is sent when this entry is first set to valid. If the first sample after this entry becomes valid is greater than or equal to **risingThreshold**, and **alarmStartupAlarm** is equal to **risingAlarm** or **risingOrFallingAlarm**, then a single rising alarm is generated. If the first sample after this entry becomes valid is less than or equal to **fallingThreshold** and **alarmStartupAlarm** is equal to **fallingAlarm** or **risingOrFallingAlarm**, then a single falling alarm is generated. For example, to set **alarmStartupAlarm** for alarm #1 to **risingOrFallingAlarm**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmStartupAlarm.1 i risingOrFallingAlarm
```

alarmRisingThreshold

A threshold for the sampled variable. When the current sampled value is greater than or equal to this threshold, and the value at the last sampling interval is less than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is greater than or equal to this threshold, and the associated **alarmStartupAlarm** is equal to **risingAlarm** or **risingOrFallingAlarm**. After a rising event is generated, another rising event cannot be generated until the sampled value falls below this threshold and reaches **alarmFallingThreshold**. For example, to set **alarmRisingThreshold** for alarm #1 to **100000**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmRisingThreshold.1 i 100000
```

alarmFallingThreshold

A threshold for the sampled variable. When the current sampled value is less than or equal to this threshold, and the value at the last sampling interval is greater than this threshold, a single event is generated. A single event is also generated if the first sample after this entry becomes valid is less than or equal to this threshold, and the associated **alarmStartupAlarm** is equal to **fallingAlarm** or **risingOrFallingAlarm**. After a falling event is generated, another falling event cannot be generated until the sampled value rises

above this threshold and reaches **alarmRisingThreshold**. For example, to set **alarmFallingThreshold** for alarm #1 to **10000**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmFallingThreshold.1 i 10000
```

alarmOwner

Any text string specified by the creating management application or the command-line interface (CLI). Typically, it is used to identify a network manager (or application) and can be used for fine access control between participating management applications.

alarmRisingEventIndex

The index of the **eventEntry** object that is used when a rising threshold is crossed. If there is no corresponding entry in **eventTable**, then no association exists. If this value is zero, no associated event is generated because zero is not a valid event index. For example, to set **alarmRisingEventIndex** for alarm #1 to **10**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmRisingEventIndex.1 i 10
```

alarmFallingEventIndex

The index of the **eventEntry** object that is used when a falling threshold is crossed. If there is no corresponding entry in **eventTable**, then no association exists. If this value is zero, no associated event is generated because zero is not a valid event index. For example, to set **alarmFallingEventIndex** for alarm #1 to **10**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community alarmFallingEventIndex.1 i 10
```

Activating a New Row in alarmTable

To activate a new row in **alarmTable**, set **alarmStatus** to **valid** using an SNMP **Set** request:

```
snmpset -Os -v2c router community alarmStatus.1 i valid
```

Modifying an Active Row in alarmTable

To modify an active row, first set **alarmStatus** to **underCreation** using an SNMP **Set** request:

```
snmpset -Os -v2c router community alarmStatus.1 i underCreation
```

Then change the row contents using an SNMP **Set** request:

```
snmpset -Os -v2c router community alarmFallingThreshold.1 i 1000
```

Finally, activate the row by setting **alarmStatus** to **valid** using an SNMP **Set** request:

```
snmpset -Os -v2c router community alarmStatus.1 i valid
```

Deactivating a Row in alarmTable

To deactivate a row in **alarmTable**, set **alarmStatus** to **invalid** using an SNMP **Set** request:

```
snmpset -Os -v2c router community alarmStatus.1 i invalid
```

Related Documentation

- Understanding RMON Alarms on page 65
- Understanding RMON Events on page 76
- Configuring an Alarm Entry and Its Attributes on page 158

Using eventTable to Log Alarms

To use **eventTable** to log alarms, perform the following tasks:

- Creating an Event Entry on page 322
- Configuring the MIB Objects on page 322
- Activating a New Row in eventTable on page 324
- Deactivating a Row in eventTable on page 324

Creating an Event Entry

The RMON **eventTable** controls the generation of notifications from the router. Notifications can be logs (entries to **logTable** and **syslogs**) or SNMP traps. Each event entry can be configured to generate any combination of these notifications (or no notification). When an event specifies that an SNMP trap is to be generated, the trap group that is used when sending the trap is specified by the value of the associated **eventCommunity** object. Consequently, the community in the trap message will match the value specified by **eventCommunity**. If nothing is configured for **eventCommunity**, a trap is sent using each trap group that has the **rmon-alarm** category configured.

Configuring the MIB Objects

Once you have created the new row in **eventTable**, set the following objects:

NOTE: **The eventType object is required. All other objects are optional.**

- eventType on page 323
- eventCommunity on page 323
- eventOwner on page 323
- eventDescription on page 323

eventType

The type of notification that the router generates when the event is triggered.

This object can be set to the following values:

- **log**—Adds the event entry to **logTable**.
- **log-and-trap**—Sends an SNMP trap and creates a log entry.
- **none**—Sends no notification.
- **snmptrap**—Sends an SNMP trap.

For example, to set **eventType** for event #1 to **log-and-trap**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community eventType.1 i log-and-trap
```

eventCommunity

The trap group that is used when generating a trap (if **eventType** is configured to send traps). If that trap group has the **rmon-alarm** trap category configured, a trap is sent to all the targets configured for that trap group. The community string in the trap matches the name of the trap group (and hence, the value of **eventCommunity**). If nothing is configured, traps are sent to each group with the **rmon-alarm** category set. For example, to set **eventCommunity** for event #1 to **boy-elroy**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community eventCommunity.1 s "boy-elroy"
```

NOTE: The **eventCommunity** object is optional. If you do not set this object, then the field is left blank.

eventOwner

Any text string specified by the creating management application or the command-line interface (CLI). Typically, it is used to identify a network manager (or application) and can be used for fine access control between participating management applications.

For example, to set **eventOwner** for event #1 to **george jetson**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community eventOwner.1 s "george jetson"
```

NOTE: The **eventOwner** object is optional. If you do not set this object, then the field is left blank.

eventDescription

Any text string specified by the creating management application or the command-line interface (CLI). The use of this string is application dependent.

For example, to set **eventDescription** for event #1 to **spacelys sprockets**, use the following SNMP **Set** request:

```
snmpset -Os -v2c router community eventDescription.1 s "spacelys sprockets"
```

NOTE: The **eventDescription** object is optional. If you do not set this object, then the field is left blank.

Activating a New Row in eventTable

To activate the new row in **eventTable**, set **eventStatus** to **valid** using an SNMP **Set** request such as:

```
snmpset -Os -v2c router community eventStatus.1 i valid
```

Deactivating a Row in eventTable

To deactivate a row in **eventTable**, set **eventStatus** to **invalid** using an SNMP **Set** request such as:

```
snmpset -Os -v2c router community eventStatus.1 i invalid
```

Related Documentation

- Understanding RMON Alarms on page 65
- Understanding RMON Events on page 76
- Configuring an Event Entry and Its Attributes on page 162

Minimum RMON Alarm and Event Entry Configuration

To enable RMON on the router, you must configure an alarm entry and an event entry. To do this, include the following statements at the **[edit snmp rmon]** hierarchy level:

```
[edit snmp rmon]
alarm index {
    rising-event-index index;
    rising-threshold integer;
    sample-type type;
    variable oid-variable;
}
event index;
```

Related Documentation

- Understanding RMON Alarms and Events Configuration on page 157
- Configuring an Alarm Entry and Its Attributes on page 158
- Configuring an Event Entry and Its Attributes on page 162

Understanding RMON for Monitoring Service Quality

Health and performance monitoring can benefit from the remote monitoring of SNMP variables by the local SNMP agents running on each router. The SNMP agents compare

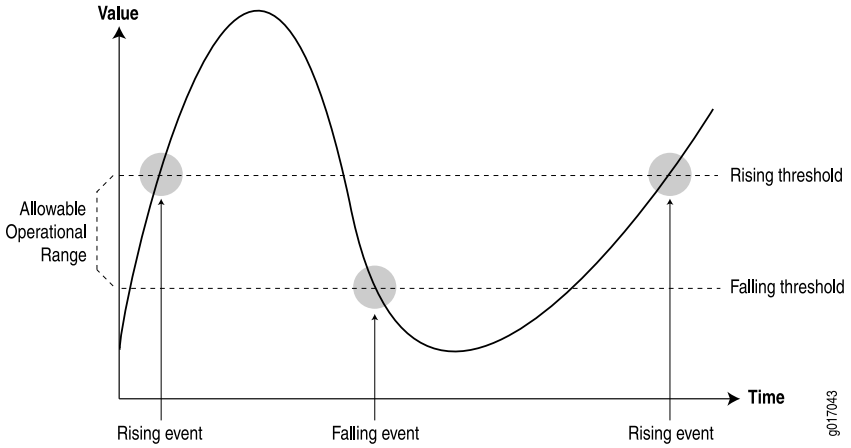
MIB values against predefined thresholds and generate exception alarms without the need for polling by a central SNMP management platform. This is an effective mechanism for proactive management, as long as the thresholds have baselines determined and set correctly. For more information, see RFC 2819, *Remote Network Monitoring MIB*.

This topic includes the following sections:

Setting Thresholds

By setting a rising and a falling threshold for a monitored variable, you can be alerted whenever the value of the variable falls outside of the allowable operational range. (See Figure 4 on page 325.)

Figure 4: Setting Thresholds



Events are only generated when the threshold is first crossed in any one direction rather than after each sample period. For example, if a rising threshold crossing event is raised, no more threshold crossing events will occur until a corresponding falling event. This considerably reduces the quantity of alarms that are produced by the system, making it easier for operations staff to react when alarms do occur.

To configure remote monitoring, specify the following pieces of information:

- The variable to be monitored (by its SNMP object identifier)
- The length of time between each inspection
- A rising threshold
- A falling threshold

- A rising event
- A falling event

Before you can successfully configure remote monitoring, you should identify what variables need to be monitored and their allowable operational range. This requires some period of baselining to determine the allowable operational ranges. An initial baseline period of at least three months is not unusual when first identifying the operational ranges and defining thresholds, but baseline monitoring should continue over the life span of each monitored variable.

RMON Command-Line Interface

Junos OS provides two mechanisms you use to control the Remote Monitoring agent on the router: command-line interface (CLI) and SNMP. To configure an RMON entry using the CLI, include the following statements at the **[edit snmp]** hierarchy level:

```
rmon {
    alarm index {
        description;
        falling-event-index;
        falling-threshold;
        intervals;
        rising-event-index;
        rising-threshold;
        sample-type (absolute-value | delta-value);
        startup-alarm (falling | rising | rising-or-falling);
        variable;
    }
    event index {
        community;
        description;
        type (log | trap | log-and-trap | none);
    }
}
```

If you do not have CLI access, you can configure remote monitoring using the SNMP Manager or management application, assuming SNMP access has been granted. (See Table 18 on page 326.) To configure RMON using SNMP, perform SNMP **Set** requests to the RMON event and alarm tables.

RMON Event Table

Set up an event for each type that you want to generate. For example, you could have two generic events, *rising* and *falling*, or many different events for each variable that is being monitored (for example, *temperature rising* event, *temperature falling* event, *firewall hit* event, *interface utilization* event, and so on). Once the events have been configured, you do not need to update them.

Table 18: RMON Event Table

Field	Description
eventDescription	Text description of this event

Table 18: RMON Event Table *(continued)*

Field	Description
eventType	Type of event (for example, **log**, **trap**, or **log** and **trap**)
eventCommunity	Trap group to which to send this event (as defined in the Junos OS configuration, which is not the same as the community)
eventOwner	Entity (for example, **manager**) that created this event
eventStatus	Status of this row (for example, **valid**, **invalid**, or **createRequest**)

RMON Alarm Table

The RMON alarm table stores the SNMP object identifiers (including their instances) of the variables that are being monitored, together with any rising and falling thresholds and their corresponding event indexes. To create an RMON request, specify the fields shown in Table 19 on page 327.

Table 19: RMON Alarm Table

Field	Description
alarmStatus	Status of this row (for example, **valid**, **invalid**, or **createRequest**)
alarmInterval	Sampling period (in seconds) of the monitored variable
alarmVariable	OID (and instance) of the variable to be monitored
alarmValue	Actual value of the sampled variable
alarmSampleType	Sample type (**absolute** or **delta** changes)
alarmStartupAlarm	Initial alarm (**rising**, **falling**, or **either**)
alarmRisingThreshold	Rising threshold against which to compare the value
alarmFallingThreshold	Falling threshold against which to compare the value
alarmRisingEventIndex	Index (row) of the rising event in the event table
alarmFallingEventIndex	Index (row) of the falling event in the event table

Both the **alarmStatus** and **eventStatus** fields are **entryStatus** primitives, as defined in RFC 2579, *Textual Conventions for SMIv2*.

Troubleshooting RMON

You troubleshoot the RMON agent, **rmopd**, that runs on the router by inspecting the contents of the Juniper Networks enterprise RMON MIB, **jnxRmon**, which provides the extensions listed in Table 20 on page 328 to the RFC 2819 **alarmTable**.

Table 20: jnxRmon Alarm Extensions

Field	Description
jnxRmonAlarmGetFailCnt	Number of times the internal **Get** request for the variable failed
jnxRmonAlarmGetFailTime	Value of **sysUpTime** when the last failure occurred
jnxRmonAlarmGetFailReason	Reason why the **Get** request failed
jnxRmonAlarmGetOkTime	Value of **sysUpTime** when the variable moved out of failure state
jnxRmonAlarmState	Status of this alarm entry

Monitoring the extensions in this table provides clues as to why remote alarms may not behave as expected.

Related Documentation

- Understanding Measurement Points, Key Performance Indicators, and Baseline Values on page 78
- Defining and Measuring Network Availability on page 328
- Measuring Health on page 334
- Measuring Performance on page 340

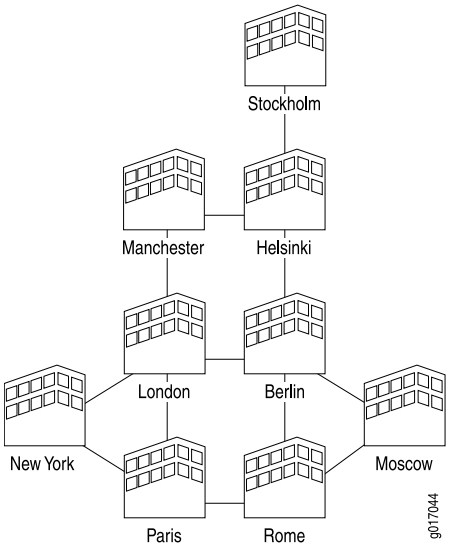
Defining and Measuring Network Availability

This topic includes the following sections:

Defining Network Availability

Availability of a service provider's IP network can be thought of as the reachability between the regional points of presence (POP), as shown in Figure 5 on page 329.

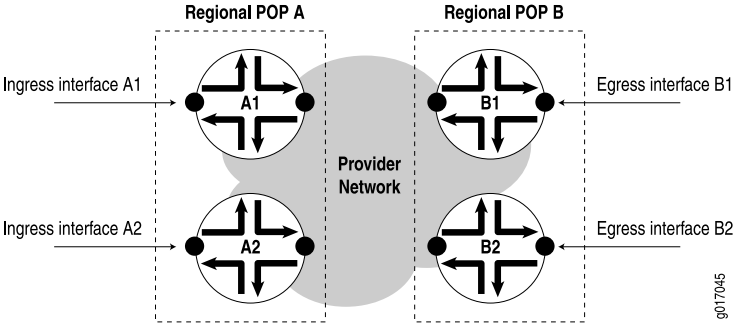
Figure 5: Regional Points of Presence



With the example above, when you use a full mesh of measurement points, where every POP measures the availability to every other POP, you can calculate the total availability of the service provider's network. This KPI can also be used to help monitor the service level of the network, and can be used by the service provider and its customers to determine if they are operating within the terms of their service-level agreement (SLA).

Where a POP may consist of multiple routers, take measurements to each router as shown in Figure 6 on page 329.

Figure 6: Measurements to Each Router



Measurements include:

- Path availability—Availability of an egress interface **B1** as seen from an ingress interface **A1**.
- Router availability—Percentage of path availability of all measured paths terminating on the router.

- POP availability—Percentage of router availability between any two regional POPs, **A** and **B**.
- Network availability—Percentage of POP availability for all regional POPs in the service provider's network.

To measure POP availability of **POP A** to **POP B** in Figure 6 on page 329, you must measure the following four paths:

Path A1 => B1
Path A1 => B2
Path A2 => B1
Path A2 => B2

Measuring availability from **POP B** to **POP A** would require a further four measurements, and so on.

A full mesh of availability measurements can generate significant management traffic. From the sample diagram above:

- Each POP has two co-located provider edge (PE) routers, each with 2xSTM1 interfaces, for a total of 18 PE routers and 36xSTM1 interfaces.
- There are six core provider (P) routers, four with 2xSTM4 and 3xSTM1 interfaces each, and two with 3xSTM4 and 3xSTM1 interfaces each.

This makes a total of 68 interfaces. A full mesh of paths between every interface is:

[**n** x (**n**–**1**)] / **2** gives [**68** x (**68**–**1**)] / **2**=**2278** paths

To reduce management traffic on the service provider's network, instead of generating a full mesh of interface availability tests (for example, from each interface to every other interface), you can measure from each router's loopback address. This reduces the number of availability measurements required to a total of one for each router, or:

[**n** x (**n**–**1**)] / **2** gives [**24** x (**24**–**1**)] / **2**=**276** measurements

This measures availability from each router to every other router.

Monitoring the SLA and the Required Bandwidth

A typical SLA between a service provider and a customer might state:

> A Point of Presence is the connection of two back-to-back provider edge routers to separate core provider routers using different links for resilience. The system is considered to be unavailable when either an entire POP becomes unavailable or for the duration of a Priority 1 fault.

An SLA availability figure of 99.999 percent for a provider's network would relate to a down time of approximately 5 minutes per year. Therefore, to measure this proactively, you would have to take availability measurements at a granularity of less than one every five minutes. With a standard size of 64 bytes per ICMP ping request, one ping test per minute would generate 7680 bytes of traffic per hour per destination, including ping responses. A full mesh of ping tests to 276 destinations would generate 2,119,680 bytes per hour, which represents the following:

- On an OC3/STM1 link of 155.52 Mbps, a utilization of 1.362 percent
- On an OC12/STM4 link of 622.08 Mbps, a utilization of 0.340 percent

With a size of 1500 bytes per ICMP ping request, one ping test per minute would generate 180,000 bytes per hour per destination, including ping responses. A full mesh of ping tests to 276 destinations would generate 49,680,000 bytes per hour, which represents the following:

- On an OC3/STM1 link, 31.94 percent utilization
- On an OC12/STM4 link, 7.986 percent utilization

Each router can record the results for every destination tested. With one test per minute to each destination, a total of 1 x 60 x 24 x 276 = 397,440 tests per day would be performed and recorded by each router. All ping results are stored in the **pingProbeHistoryTable** (see RFC 2925) and can be retrieved by an SNMP performance reporting application (for example, service performance management software from InfoVista, Inc., or Concord Communications, Inc.) for post processing. This table has a maximum size of 4,294,967,295 rows, which is more than adequate.

Measuring Availability

There are two methods you can use to measure availability:

- Proactive—Availability is automatically measured as often as possible by an operational support system.
- Reactive—Availability is recorded by a Help desk when a fault is first reported by a user or a fault monitoring system.

This section discusses real-time performance monitoring as a proactive monitoring solution.

Real-Time Performance Monitoring

Juniper Networks provides a real-time performance monitoring (RPM) service to monitor real-time network performance. Use the J-Web Quick Configuration feature to configure real-time performance monitoring parameters used in real-time performance monitoring tests. (J-Web Quick Configuration is a browser-based GUI that runs on Juniper Networks routers. For more information, see the *J-Web Interface User Guide*.)

Configuring Real-Time Performance Monitoring

Some of the most common options you can configure for real-time performance monitoring tests are shown in Table 21 on page 331.

Table 21: Real-Time Performance Monitoring Configuration Options

Field	Description
Request Information	

Table 21: Real-Time Performance Monitoring Configuration Options *(continued)*

Field	Description
Probe Type	Type of probe to send as part of the test. Probe types can be: • **http-get** • **http-get-metadata** • **icmp-ping** • **icmp-ping-timestamp** • **tcp-ping** • **udp-ping**
Interval	Wait time (in seconds) between each probe transmission. The range is 1 to 255 seconds.
Test Interval	Wait time (in seconds) between tests. The range is 0 to 86400 seconds.
Probe Count	Total number of probes sent for each test. The range is 1 to 15 probes.
Destination Port	TCP or UDP port to which probes are sent. Use number 7—a standard TCP or UDP port number—or select a port number from 49152 through 65535.
DSCP Bits	Differentiated Services code point (DSCP) bits. This value must be a valid 6-bit pattern. The default is 000000.
Data Size	Size (in bytes) of the data portion of the ICMP probes. The range is 0 to 65507 bytes.
Data Fill	Contents of the data portion of the ICMP probes. Contents must be a hexadecimal value. The range is 1 to 800h.
Maximum Probe Thresholds	
Successive Lost Probes	Total number of probes that must be lost successively to trigger a probe failure and generate a system log message. The range is 0 to 15 probes.
Lost Probes	Total number of probes that must be lost to trigger a probe failure and generate a system log message. The range is 0 to 15 probes.
Round Trip Time	Total round-trip time (in microseconds) from the Services Router to the remote server, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Jitter	Total jitter (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.

Table 21: Real-Time Performance Monitoring Configuration Options *(continued)*

Field	Description
Standard Deviation	Maximum allowable standard deviation (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Egress Time	Total one-way time (in microseconds) from the router to the remote server, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Ingress Time	Total one-way time (in microseconds) from the remote server to the router, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Jitter Egress Time	Total outbound-time jitter (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Jitter Ingress Time	Total inbound-time jitter (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Egress Standard Deviation	Maximum allowable standard deviation of outbound times (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.
Ingress Standard Deviation	Maximum allowable standard deviation of inbound times (in microseconds) for a test, which, if exceeded, triggers a probe failure and generates a system log message. The range is 0 to 60,000,000 microseconds.

Displaying Real-Time Performance Monitoring Information

For each real-time performance monitoring test configured on the router, monitoring information includes the round-trip time, jitter, and standard deviation. To view this information, select **Monitor > RPM** in the J-Web interface, or enter the **show services rpm** command-line interface (CLI) command.

To display the results of the most recent real-time performance monitoring probes, enter the **show services rpm probe-results** CLI command:

```
user@host> show services rpm probe-results
Owner: p1, Test: t1
   Target address: 10.8.4.1, Source address: 10.8.4.2, Probe type: icmp-ping
   Destination interface name: lt-0/0/0.0
   Test size: 10 probes
   Probe results:
      Response received, Sun Jul 10 19:07:34 2005
      Rtt: 50302 usec
   Results over current test:
```

```
    Probes sent: 2, Probes received: 1, Loss percentage: 50
    Measurement: Round trip time
      Minimum: 50302 usec, Maximum: 50302 usec, Average: 50302 usec,
      Jitter: 0 usec, Stddev: 0 usec
Results over all tests:
    Probes sent: 2, Probes received: 1, Loss percentage: 50
    Measurement: Round trip time
      Minimum: 50302 usec, Maximum: 50302 usec, Average: 50302 usec,
      Jitter: 0 usec, Stddev: 0 usec
```

Related Documentation

- Understanding Measurement Points, Key Performance Indicators, and Baseline Values on page 78
- Understanding RMON for Monitoring Service Quality on page 324
- Measuring Health on page 334
- Measuring Performance on page 340

Measuring Health

You can monitor health metrics reactively by using fault management software such as SMARTS InCharge, Micromuse Netcool Omnibus, or Concord Live Exceptions. We recommend that you monitor the health metrics shown in Table 22 on page 334.

Table 22: Health Metrics

Metric:	Errors in
Description	Number of inbound packets that contained errors, preventing them from being delivered
MIB name	IF-MIB (RFC 2233)
Variable name	**ifInErrors**
Variable OID	.1.3.6.1.31.2.2.1.14
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	Logical interfaces
Metric:	Errors out
Description	Number of outbound packets that contained errors, preventing them from being transmitted
MIB name	IF-MIB (RFC 2233)
Variable name	**ifOutErrors**

Table 22: Health Metrics *(continued)*

Variable OID	.1.3.6.1.31.2.2.1.20
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	Logical interfaces
Metric:	Discards in
Description	Number of inbound packets discarded, even though no errors were detected
MIB name	IF-MIB (RFC 2233)
Variable name	**ifInDiscards**
Variable OID	.1.3.6.1.31.2.2.1.13
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	Logical interfaces
Metric:	Unknown protocols
Description	Number of inbound packets discarded because they were of an unknown protocol
MIB name	IF-MIB (RFC 2233)
Variable name	**ifInUnknownProtos**
Variable OID	.1.3.6.1.31.2.2.1.15
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	Logical interfaces
Metric:	Interface operating status
Description	Operational status of an interface
MIB name	IF-MIB (RFC 2233)
Variable name	**ifOperStatus**

Table 22: Health Metrics *(continued)*

Variable OID	.1.3.6.1.31.2.2.1.8
Frequency (mins)	15
Allowable range	1 (up)
Managed objects	Logical interfaces
Metric:	Label Switched Path (LSP) state
Description	Operational state of an MPLS label-switched path
MIB name	MPLS-MIB
Variable name	**mplsLspState**
Variable OID	mplsLspEntry.2
Frequency (mins)	60
Allowable range	2 (up)
Managed objects	All label-switched paths in the network
Metric:	Component operating status
Description	Operational status of a router hardware component
MIB name	JUNIPER-MIB
Variable name	**jnxOperatingState**
Variable OID	.1.3.6.1.4.1.2636.1.13.1.6
Frequency (mins)	60
Allowable range	2 (running) or 3 (ready)
Managed objects	All components in each Juniper Networks router
Metric:	Component operating temperature
Description	Operational temperature of a hardware component, in Celsius
MIB name	JUNIPER-MIB
Variable name	**jnxOperatingTemp**
Variable OID	.1.3.6.1.4.1.2636.1.13.1.7

Table 22: Health Metrics *(continued)*

Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All components in a chassis
Metric:	System up time
Description	Time, in milliseconds, that the system has been operational.
MIB name	MIB-2 (RFC 1213)
Variable name	**sysUpTime**
Variable OID	.1.3.6.1.1.3
Frequency (mins)	60
Allowable range	Increasing only (decrement indicates a restart)
Managed objects	All routers
Metric:	No IP route errors
Description	Number of packets that could not be delivered because there was no IP route to their destination.
MIB name	MIB-2 (RFC 1213)
Variable name	**ipOutNoRoutes**
Variable OID	ip.12
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	Each router
Metric:	Wrong SNMP community names
Description	Number of incorrect SNMP community names received
MIB name	MIB-2 (RFC 1213)
Variable name	**snmpInBadCommunityNames**
Variable OID	snmp.4

Table 22: Health Metrics *(continued)*

Frequency (hours)	24
Allowable range	To be baselined
Managed objects	Each router
Metric:	SNMP community violations
Description	Number of valid SNMP communities used to attempt invalid operations (for example, attempting to perform SNMP **Set** requests)
MIB name	MIB-2 (RFC 1213)
Variable name	**snmpInBadCommunityUses**
Variable OID	snmp.5
Frequency (hours)	24
Allowable range	To be baselined
Managed objects	Each router
Metric:	Redundancy switchover
Description	Total number of redundancy switchovers reported by this entity
MIB name	JUNIPER-MIB
Variable name	**jnxRedundancySwitchoverCount**
Variable OID	jnxRedundancyEntry.8
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All Juniper Networks routers with redundant Routing Engines
Metric:	FRU state
Description	Operational status of each field-replaceable unit (FRU)
MIB name	JUNIPER-MIB
Variable name	**jnxFruState**
Variable OID	jnxFruEntry.8

Table 22: Health Metrics *(continued)*

Frequency (mins)	15
Allowable range	2 through 6 for ready/online states. See **jnxFruOfflineReason** in the event of a FRU failure.
Managed objects	All FRUs in all Juniper Networks routers.
Metric:	Rate of tail-dropped packets
Description	Rate of tail-dropped packets per output queue, per forwarding class, per interface.
MIB name	JUNIPER-COS-MIB
Variable name	**jnxCosIfqTailDropPktRate**
Variable OID	jnxCosIfqStatsEntry.12
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	For each forwarding class per interface in the provider network, when CoS is enabled.
Metric:	Interface utilization: octets received
Description	Total number of octets received on the interface, including framing characters.
MIB name	IF-MIB
Variable name	**ifInOctets**
Variable OID	.1.3.6.1.2.1.2.2.1.10.x
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All operational interfaces in the network
Metric:	Interface utilization: octets transmitted
Description	Total number of octets transmitted out of the interface, including framing characters.
MIB name	IF-MIB
Variable name	**ifOutOctets**

Table 22: Health Metrics *(continued)*

Variable OID	.1.3.6.1.2.1.2.2.1.16.x
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All operational interfaces in the network

NOTE: Byte counts vary depending on interface type, encapsulation used and PIC supported. For example, with vlan-ccc encapsulation on a 4xFE, GE, or GE 1Q PIC, the byte count includes framing and control word overhead. (See Table 23 on page 340.)

Table 23: Counter Values for vlan-ccc Encapsulation

PIC Type	Encapsulation	input (Unit Level)	Output (Unit Level)	SNMP
4xFE	vlan-ccc	Frame (no frame check sequence [FCS])	Frame (including FCS and control word)	ifInOctets, ifOutOctets
GE	vlan-ccc	Frame (no FCS)	Frame (including FCS and control word)	ifInOctets, ifOutOctets
GE IQ	vlan-ccc	Frame (no FCS)	Frame (including FCS and control word)	ifInOctets, ifOutOctets

SNMP traps are also a good mechanism to use for health management. For more information, see "“Standard SNMP Traps Supported on Devices Running Junos OS” on page 63" and "“Juniper Networks Enterprise-Specific SNMP Traps” on page 63" in the *SNMP MIBs and Traps Reference*.

Related Documentation

- Understanding Measurement Points, Key Performance Indicators, and Baseline Values on page 78
- Understanding RMON for Monitoring Service Quality on page 324
- Defining and Measuring Network Availability on page 328
- Measuring Performance on page 340

Measuring Performance

The performance of a service provider's network is usually defined as how well it can support services, and is measured with metrics such as delay and utilization. We suggest that you monitor the following performance metrics using applications such as InfoVista Service Performance Management or Concord Network Health (see Table 24 on page 341).

Table 24: Performance Metrics

Metric:	Average delay
Description	Average round-trip time (in milliseconds) between two measurement points.
MIB name	DISMAN-PING-MIB (RFC 2925)
Variable name	**pingResultsAverageRtt**
Variable OID	pingResultsEntry.6
Frequency (mins)	15 (or depending upon ping test frequency)
Allowable range	To be baselined
Managed objects	Each measured path in the network
Metric:	Interface utilization
Description	Utilization percentage of a logical connection.
MIB name	IF-MIB
Variable name	(**ifInOctets** & **ifOutOctets**) * 8 / **ifSpeed**
Variable OID	ifTable entries
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All operational interfaces in the network
Metric:	Disk utilization
Description	Utilization of disk space within the Juniper Networks router
MIB name	HOST-RESOURCES-MIB (RFC 2790)
Variable name	**hrStorageSize – hrStorageUsed**
Variable OID	hrStorageEntry.5 – hrStorageEntry.6
Frequency (mins)	1440
Allowable range	To be baselined
Managed objects	All Routing Engine hard disks

Table 24: Performance Metrics *(continued)*

Metric:	Memory utilization
Description	Utilization of memory on the Routing Engine and FPC.
MIB name	JUNIPER-MIB (Juniper Networks enterprise Chassis MIB)
Variable name	**jnxOperatingHeap**
Variable OID	Table for each component
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All Juniper Networks routers
Metric:	CPU load
Description	Average utilization over the past minute of a CPU.
MIB name	JUNIPER-MIB (Juniper Networks enterprise Chassis MIB)
Variable name	**jnxOperatingCPU**
Variable OID	Table for each component
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All Juniper Networks routers
Metric:	LSP utilization
Description	Utilization of the MPLS label-switched path.
MIB name	MPLS-MIB
Variable name	**mplsPathBandwidth / (mplsLspOctets * 8)**
Variable OID	mplsLspEntry.21 and mplsLspEntry.3
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	All label-switched paths in the network
Metric:	Output queue size

Table 24: Performance Metrics *(continued)*

Description	Size, in packets, of each output queue per forwarding class, per interface.
MIB name	JUNIPER-COS-MIB
Variable name	**jnxCosIfqQedPkts**
Variable OID	jnxCosIfqStatsEntry.3
Frequency (mins)	60
Allowable range	To be baselined
Managed objects	For each forwarding class per interface in the network, once CoS is enabled.

This section includes the following topics:

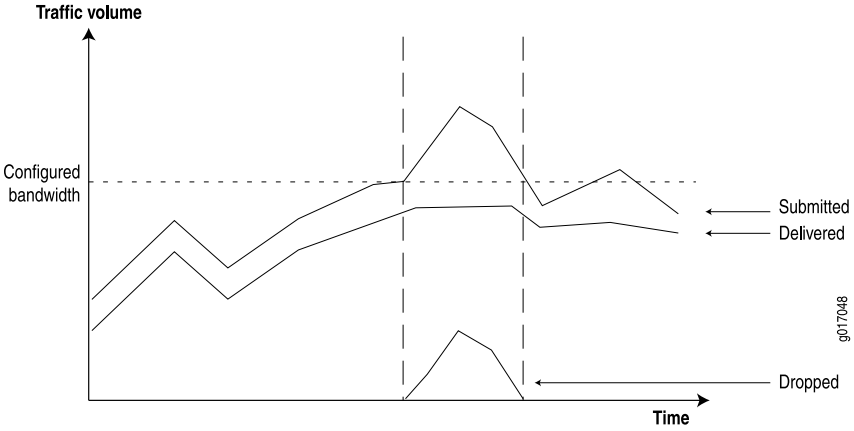
Measuring Class of Service

You can use class-of-service (CoS) mechanisms to regulate how certain classes of packets are handled within your network during times of peak congestion. Typically you must perform the following steps when implementing a CoS mechanism:

- Identify the type of packets that is applied to this class. For example, include all customer traffic from a specific ingress edge interface within one class, or include all packets of a particular protocol such as voice over IP (VoIP).
- Identify the required deterministic behavior for each class. For example, if VoIP is important, give VoIP traffic the highest priority during times of network congestion. Conversely, you can downgrade the importance of Web traffic during congestion, as it may not impact customers too much.

With this information, you can configure mechanisms at the network ingress to monitor, mark, and police traffic classes. Marked traffic can then be handled in a more deterministic way at egress interfaces, typically by applying different queuing mechanisms for each class during times of network congestion. You can collect information from the network to provide customers with reports showing how the network is behaving during times of congestion. (See Figure 7 on page 344.)

Figure 7: Network Behavior During Congestion



To generate these reports, routers must provide the following information:

- Submitted traffic—Amount of traffic received per class.
- Delivered traffic—Amount of traffic transmitted per class.
- Dropped traffic—Amount of traffic dropped because of CoS limits.

The following section outlines how this information is provided by Juniper Networks routers.

Inbound Firewall Filter Counters per Class

Firewall filter counters are a very flexible mechanism you can use to match and count inbound traffic per class, per interface. For example:

```
firewall {
   filter f1 {
      term t1 {
         from {
            dscp af11;
         }
         then {
            # Assured forwarding class 1 drop profile 1 count inbound-af11;
            accept;
         }
      }
   }
}
```

For example, Table 25 on page 344 shows additional filters used to match the other classes.

Table 25: Inbound Traffic Per Class

DSCP Value	Firewall Match Condition	Description
10	af11	Assured forwarding class 1 drop profile 1
12	af12	Assured forwarding class 1 drop profile 2

Table 25: Inbound Traffic Per Class *(continued)*

DSCP Value	Firewall Match Condition	Description
18	af21	Best effort class 2 drop profile 1
20	af22	Best effort class 2 drop profile 2
26	af31	Best effort class 3 drop profile 1

Any packet with a CoS DiffServ code point (DSCP) conforming to RFC 2474 can be counted in this way. The Juniper Networks enterprise-specific Firewall Filter MIB presents the counter information in the variables shown in Table 26 on page 345.

Table 26: Inbound Counters

Indicator Name	Inbound Counters
MIB	**jnxFirewalls**
Table	**jnxFirewallCounterTable**
Index	**jnxFWFilter.jnxFWCounter**
Variables	**jnxFWCounterPacketCount** **jnxFWCounterByteCount**
Description	Number of bytes being counted pertaining to the specified firewall filter counter
SNMP version	SNMPv2

This information can be collected by any SNMP management application that supports SNMPv2. Products from vendors such as Concord Communications, Inc., and InfoVista, Inc., provide support for the Juniper Networks Firewall MIB with their native Juniper Networks device drivers.

Monitoring Output Bytes per Queue

You can use the Juniper Networks enterprise ATM CoS MIB to monitor outbound traffic, per virtual circuit forwarding class, per interface. (See Table 27 on page 345.)

Table 27: Outbound Counters for ATM Interfaces

Indicator Name	Outbound Counters
MIB	**JUNIPER-ATM-COS-MIB**
Variable	**jnxCosAtmVcQstatsOutBytes**
Index	**ifIndex.atmVclVpi.atmVclVci.jnxCosFcId**

Table 27: Outbound Counters for ATM Interfaces *(continued)*

Indicator Name	Outbound Counters
Description	Number of bytes belonging to the specified forwarding class that were transmitted on the specified virtual circuit.
SNMP version	SNMPv2

Non-ATM interface counters are provided by the Juniper Networks enterprise-specific CoS MIB, which provides information shown in Table 28 on page 346.

Table 28: Outbound Counters for Non-ATM Interfaces

Indicator Name	Outbound Counters
MIB	**JUNIPER-COS-MIB**
Table	**jnxCosIfqStatsTable**
Index	**jnxCosIfqIfIndex.jnxCosIfqFc**
Variables	**jnxCosIfqTxedBytes** **jnxCosIfqTxedPkts**
Description	Number of transmitted bytes or packets per interface per forwarding class
SNMP version	SNMPv2

Dropped Traffic

You can calculate the amount of dropped traffic by subtracting the outbound traffic from the incoming traffic:

Dropped = Inbound Counter – Outbound Counter

You can also select counters from the CoS MIB, as shown in Table 29 on page 346.

Table 29: Dropped Traffic Counters

Indicator Name	Dropped Traffic
MIB	**JUNIPER-COS-MIB**
Table	**jnxCosIfqStatsTable**
Index	**jnxCosIfqIfIndex.jnxCosIfqFc**
Variables	**jnxCosIfqTailDropPkts** **jnxCosIfqTotalRedDropPkts**

Table 29: Dropped Traffic Counters *(continued)*

Indicator Name	Dropped Traffic
Description	The number of tail-dropped or RED-dropped packets per interface per forwarding class
SNMP version	SNMPv2

Related Documentation

- Understanding Measurement Points, Key Performance Indicators, and Baseline Values on page 78
- Understanding RMON for Monitoring Service Quality on page 324
- Defining and Measuring Network Availability on page 328
- Measuring Health on page 334

PART 4

Troubleshooting

CHAPTER 18

Best Practices

Junos OS SNMP FAQs Overview

SNMP enables users to monitor network devices from a central location. Many network management systems (NMS) are based on SNMP, and support for this protocol is a key feature of most network devices.

Juniper Networks provides many different platforms that support SNMP on Junos OS. Junos OS includes an onboard SNMP agent that provides remote management applications with access to detailed information about the devices on the network.

A typical SNMP implementation contains three components:

- Managed devices – Such as routers and switches.
- SNMP agent – Process that resides on a managed device and communicates with the NMS.
- NMS – A combination of hardware and software used to monitor and administer the network; network device that runs SNMP manager software. Also referred to as an SNMP manager.

The SNMP agent exchanges network management information with the SNMP manager (NMS). The agent responds to requests for information and actions from the manager. The SNMP manager collects information about network connectivity, activity, and events by polling managed devices.

SNMP implementation in Junos OS uses a master SNMP agent (known as an SNMP process or snmpd) that resides on the managed device. Various subagents reside on different modules of Junos OS as well (such as the Routing Engine), and these subagents are managed by the snmpd.

Related Documentation

- Junos OS SNMP FAQs on page 352

Junos OS SNMP FAQs

This Frequently Asked Questions technology overview covers these SNMP-related areas:

Junos OS SNMP Support FAQs

This section presents frequently asked questions and answers related to SNMP support on Junos OS.

Which SNMP versions does Junos OS support?

Junos OS supports SNMP version 1 (SNMPv1), version 2 (SNMPv2c), and version 3 (SNMPv3). By default, SNMP is disabled on a Juniper Networks device. To enable SNMP, see the instructions in *Configuring SNMP on Devices Running Junos OS* in the *Best Practices SNMP-Based Network Management on Devices Running Junos OS* document.

Which ports (sockets) does SNMP use?

The default port for SNMP queries is port 161. The default port for SNMP traps and informs is port 162. The ports used by SNMP are configurable, and you can configure your system to use ports other than the defaults.

Is SNMP support different among the Junos OS platforms?

No, SNMP support is not different among the Junos OS platforms. SNMP configuration, interaction, and behavior are the same on any Junos OS device. The only difference that might occur across platforms is MIB support. See "Junos OS MIBs FAQs" on page 353 for more information about MIB support on the Junos OS platforms.

See also the *Junos OS Network Management Configuration Guide* for a list of MIBs that are supported across the Junos OS platforms.

Does Junos OS support the user-based security model (USM)?

Yes, Junos OS supports USM as part of its support for SNMPv3. SNMPv3 contains more security measures than previous versions of SNMP, including providing a defined USM. SNMPv3 USM provides message security through data integrity, data origin authentication, message replay protection, and protection against disclosure of the message payload.

Does Junos OS support the view-based access control model (VACM)?

Yes, Junos OS supports VACM as part of its support for SNMPv3. SNMPv3 contains more security measures than previous versions of SNMP, including providing a defined VACM. SNMPv3 VACM determines whether a specific type of access (read or write) to the management information is allowed.

Does Junos OS support SNMP informs?

Yes, Junos OS supports SNMP informs as part of its support for SNMPv3. SNMP informs are confirmed notifications sent from SNMP agents to SNMP managers when significant events occur on a network device. When an SNMP manager receives an inform, it sends a response to the sender to verify receipt of the inform.

Can I provision or configure a device using SNMP on Junos OS?

No, provisioning or configuring a device using SNMP is not allowed on Junos OS.

Related Documentation

- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Configuration FAQs on page 361
- SNMPv3 FAQs on page 365
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Traps and Informs FAQs on page 369
- SNMP Support for Routing Instances FAQs on page 376
- SNMP Counters FAQs on page 377

Junos OS MIBs FAQs

This section presents frequently asked questions and answers related to Junos OS MIBs.

What is a MIB?

A management information base (MIB) is a table of definitions for managed objects in a network device. MIBs are used by SNMP to maintain standard definitions of all of the components and their operating conditions within a network device. Each object in the MIB has an identifying code called an object identifier (OID).

MIBs are either standard or enterprise-specific. Standard MIBs are created by the Internet Engineering Task Force (IETF) and documented in various RFCs. Enterprise-specific MIBs are developed and supported by a specific equipment manufacturer.

For a list of supported standard MIBs, see "Standard SNMP MIBs Supported by Junos OS" on page 19 in the *Junos OS SNMP MIBs and Traps Reference* document.

For a list of Juniper Networks enterprise-specific MIBs, see "Juniper Networks Enterprise-Specific MIBs" on page 35 in the *Junos OS SNMP MIBs and Traps Reference* document.

Do MIB files reside on the Junos OS devices?

No, MIB files do not reside on the Junos OS devices. You must download the MIB files from the Juniper Networks Technical Publications page for the required Junos OS release: http://www.juniper.net/techpubs/en_US/release-independent/junos/mibs/mibs.html .

How do I compile and load the Junos OS MIBs onto an SNMP manager or NMS?

For your network management systems (NMSs) to identify and understand the MIB objects used by Junos OS, you must first load the MIB files to your NMS using a MIB compiler. A MIB compiler is a utility that parses the MIB information, such as the MIB object names, IDs, and data types for the NMS.

You can download the Junos OS MIB package from the Enterprise-Specific MIBs and Traps section at http://www.juniper.net/techpubs/en_US/release-independent/junos/mibs/mibs.html or http://www.juniper.net/techpubs/software/junos/index.html .

The Junos OS MIB package has two folders: **StandardMibs**, containing standard MIBs supported on Juniper Networks devices, and **JuniperMibs**, containing Juniper Networks enterprise-specific MIBs. You *must* have the required standard MIBs downloaded and decompressed before downloading any enterprise-specific MIBs. There might be dependencies that require a particular standard MIB to be present on the compiler before loading a particular enterprise-specific MIB.

The Junos OS MIB package is available in **.zip** and **.tar** formats. Download the format appropriate for your requirements.

Use the following steps to load MIB files for devices running Junos OS:

1. Navigate to the appropriate Juniper Networks software download page and locate the **Enterprise MIBs** link under the **Enterprise-Specific MIBs and Traps** section.

   NOTE: Although the link is titled Enterprise MIBs, both standard MIBs and enterprise-specific MIBs are available for download from this location.

2. Click the **TAR** or **ZIP** link to download the Junos OS MIB package.
3. Decompress the file (**.tar** or **.zip**) using an appropriate utility.

   NOTE: Some commonly used MIB compilers are preloaded with standard MIBs. You can skip Step 4 and Step 5 and proceed to Step 6 if you already have the standard MIBs loaded on your system.

4. Load the standard MIB files from the **StandardMibs** folder.

   Load the files in the following order:

   a. mib-SNMPv2-SMI.txt

   b. mib-SNMPv2-TC.txt

   c. mib-IANAifType-MIB.txt

d. mib-IANA-RTPROTO-MIB.txt

e. mib-rfc1907.txt

f. mib-rfc2011a.txt

g. mib-rfc2012a.txt

h. mib-rfc2013a.txt

i. mib-rfc2863a.txt

5. Load any remaining standard MIB files.

NOTE: You must follow the order specified in this procedure, and ensure that all standard MIBs are loaded before you load the enterprise-specific MIBs. There might be dependencies that require a particular standard MIB to be present on the compiler before loading a particular enterprise-specific MIB. Dependencies are listed in the IMPORT section of the MIB file.

6. After loading the standard MIBs, load the Juniper Networks enterprise-specific SMI MIB, **mib-jnx-smi.txt**, and the following optional SMI MIBs based on your requirements:

   - mib-jnx-exp.txt—(Recommended) for Juniper Networks experimental MIB objects
   - mib-jnx-js-smi.txt—(Optional) for Juniper Security MIB tree objects
   - mib-jnx-ex-smi.txt—(Optional) for EX Series Ethernet Switches

7. Load any remaining desired enterprise-specific MIBs from the **JuniperMibs** folder.

TIP: While loading a MIB file, if the compiler returns an error message indicating that any of the objects are undefined, open the MIB file using a text editor and ensure that all the MIB files listed in the IMPORT section are loaded on the compiler. If any of the MIB files listed in the IMPORT section are not loaded on the compiler, load the missing file or files first, then try to load the MIB file that failed.

The system might return an error if files are not loaded in a particular order.

What is SMI?

Structure of Management Information Version (SMI) is a subset of Abstract Syntax Notation One (ASN.1), which describes the structure of objects. SMI is the notation syntax, or “grammar”, that is the standard for writing MIBs.

Which versions of SMI does Junos OS support?

The Junos OS supports SMIv1 for SNMPv1 MIBs, and SMIv2 for SNMPv2c and enterprise MIBs.

Does Junos OS support MIB II?

Yes, Junos OS supports MIB II, the second version of the MIB standard.

The features of MIB II include:

- Additions that reflect new operational requirements.
- Backward compatibility with the original MIBs and SNMP.
- Improved support for multiprotocol entities.
- Improved readability.

Refer to the relevant release documentation for a list of MIBs that are supported. Go to http://www.juniper.net/techpubs/software/junos/index.html .

Are the same MIBs supported across all Juniper Networks devices?

There are some common MIBs supported by all the Junos OS devices, such as the Interface MIB (ifTable), System MIB, and Chassis MIB. Some MIBs are supported only by functionalities on specific platforms. For example, the Bridge MIB is supported on the EX Series Ethernet Switches and the SRX Series Services Gateways for the branch.

What is the system object identifier (SYSOID) of a device? How do I determine the SYSOID of my device?

The jnx-chas-defines (Chassis Definitions for Router Model) MIB has a **jnxProductName** branch for every Junos OS device. The system object ID of a device is identical to the object ID of the **jnxProductName** for the platform. For example, for an M7i Multiservice Edge Router, the jnxProductNameM7i is .1.3.6.1.4.1.2636.1.1.1.2.10 in the jnxProductName branch, which is identical to the SYSOID of the M7i (.1.3.6.1.4.1.2636.1.1.1.2.10).

How can I determine if a MIB is supported on a platform? How can I determine which MIBs are supported by a device?

MIBs device and platform support is listed on the Junos OS Technical Documentation index page. Go to http://www.juniper.net/techpubs/software/junos/ and select your version or release of Junos OS. Navigate to the *SNMP MIBs and Traps Reference*. The *SNMP MIBs and Traps Reference* specifies which MIBs are supported on the different platforms.

What can I do if the MIB OID query is not responding?

There can be various reasons why the MIB OID query stops responding. One reason could be that the MIB itself is unresponsive. To verify that the MIB responds, use the **show snmp mib walk | get *MIB name* | *MIB OID*** command:

- If the MIB responds, the communication issue exists between the SNMP master and SNMP agent. Possible reasons for this issue include network issues, an incorrect community configuration, an incorrect SNMP configuration, and so on.
- If the MIB does not respond, enable SNMP **traceoptions** to log PDUs and errors. All incoming and outgoing SNMP PDUs are logged. Check the **traceoptions** output to see if there are any errors.

If you continue to have problems with the MIB OID query, technical product support is available through the Juniper Networks Technical Assistance Center (JTAC).

What is the enterprise branch number for Junos OS?

The enterprise branch number for Junos OS is 2636. Enterprise branch numbers are used in SNMP MIB configurations, and they are also known as SMI network management private enterprise codes.

Which MIB displays the hardware and chassis details on a Juniper Networks device?

The Chassis MIB (jnxchassis.mib) displays the hardware and chassis details for each Juniper Networks device. It provides information about the router and its components. The Chassis MIB objects represent each component and its status.

For more information about enterprise-specific Chassis MIBs, see *Chassis MIBs* in the *Junos OS SNMP MIBs and Traps Reference* document.

Does Junos OS support the Entity MIB?

No, Junos OS does not support the Entity MIB, which is designed to identify physical and logical elements of a managed device. Instead, Junos OS supports the enterprise-specific Chassis MIB to identify the chassis components on the device.

Which MIB objects can I query to determine the CPU and memory utilization of the Routing Engine, Flexible PIC Concentrator (FPC), and PIC components on a device?

Query the Chassis MIB objects **jnxOperatingMemory**, **jnxOperatingtBuffer**, and **jnxOperatingCPU** to find out the CPU and memory utilization of the hardware components of a device.

Is the interface index (ifIndex) persistent?

For the Junos OS Release 10.0 and earlier, the ifIndex is persistent when reboots occur if the Junos OS version remains the same, meaning the values assigned to the interfaces in the ifIndex do not change. When there is a software upgrade, the device tries to keep the ifIndex persistent on a best effort basis.

For the Junos OS Release 10.1 and later, the ifIndex is persistent on all platforms, except for the EX4200 virtual chassis because it can have over 500 interfaces.

Is it possible to set the ifAdminStatus?

SNMP is not allowed to set the ifAdminStatus.

Which MIB objects support SNMP set operations?

The Junos OS SNMP set operations are supported in the following MIB tables and variables:

- snmpCommunityTable
- eventTable
- alarmTable

- snmpTargetAddrExtTable
- jnxPingCtlTable
- pingCtlTable
- traceRouteCtlTable
- jnxTraceRouteCtlTable
- sysContact.0
- sysName.0
- sysLocation.0
- pingMaxConcurrentRequests.0
- traceRouteMaxConcurrentRequests.0
- usmUserSpinLock
- usmUserOwnAuthKeyChange
- usmUserPublic
- vacmSecurityToGroupTable (vacmGroupName, vacmSecurityToGroupStorageType, and vacmSecurityToGroupStatus)
- vacmAccessTable (vacmAccessContextMatch, vacmAccessReadViewName, vacmAccessWriteViewName, vacmAccessNotifyViewName, vacmAccessStorageType, and vacmAccessStatus)
- vacmViewSpinLock
- vacmViewTreeFamilyTable (vacmViewTreeFamilyMask, vacmViewTreeFamilyType, vacmViewTreeFamilyStorageType, and vacmViewTreeFamilyStatus)

Does Junos OS support remote monitoring (RMON)?

Yes, Junos OS supports RMON as defined in RFC 2819, *Remote Network Monitoring Management Information Base*. However, remote monitoring version 2 (RMON 2) is not supported.

Can I use SNMP to determine the health of the processes running on the Routing Engine?

Yes, you can use SNMP to determine the health of the Routing Engine processes by configuring the health monitoring feature. On Juniper Networks devices, RMON alarms and events provide much of the infrastructure needed to reduce the polling overhead from the NMS. However, you must set up the NMS to configure specific MIB objects into RMON alarms. This often requires device-specific expertise and customizing the monitoring application. Additionally, some MIB object instances that need monitoring are set only at initialization, or they change at runtime and cannot be configured in advance.

To address these issues, the health monitor extends the RMON alarm infrastructure to provide predefined monitoring for a selected set of object instances, such as file system

usage, CPU usage, and memory usage, and includes support for unknown or dynamic object instances, such as Junos OS software processes.

To display the health monitoring configuration, use the **show snmp health-monitor** command:

```
user@host> show snmp health-monitor
interval 300;
rising-threshold 90;
falling-threshold 80;
```

When you configure the health monitor, monitoring information for certain object instances is available, as shown in Table 30 on page 359.

Table 30: Monitored Object Instances

Object	Description
jnxHrStoragePercentUsed.1	Monitors the following file system on the router or switch: /dev/ad0s1a: This is the root file system mounted on **/**.
jnxHrStoragePercentUsed.2	Monitors the following file system on the router or switch: /dev/ad0s1e: This is the configuration file system mounted on **/config**.
jnxOperatingCPU (RE0) jnxOperatingCPU (RE1)	Monitor CPU usage for Routing Engines RE0 and RE1. The index values assigned to the Routing Engines depend on whether the Chassis MIB uses a zero-based or a ones-based indexing scheme. Because the indexing scheme is configurable, the correct index is determined whenever the router is initialized and when there is a configuration change. If the router or switch has only one Routing Engine, the alarm entry monitoring RE1 is removed after five failed attempts to obtain the CPU value.
jnxOperatingBuffer (RE0) jnxOperatingBuffer (RE1)	Monitor the amount of memory available on Routing Engines RE0 and RE1. Because the indexing of this object is identical to that used for jnxOperatingCPU, index values are adjusted depending on the indexing scheme used in the Chassis MIB. As with jnxOperatingCPU, the alarm entry monitoring RE1 is removed if the router or switch has only one Routing Engine.
sysApplElmtRunCPU	Monitors the CPU usage for each Junos OS software process. Multiple instances of the same process are monitored and indexed separately.
sysApplElmtRunMemory	Monitors the memory usage for each Junos OS software process. Multiple instances of the same process are monitored and indexed separately.

The system log entries generated for any health monitor events, such as thresholds crossed and errors, have a corresponding **HEALTHMONITOR** tag rather than a generic **SNMPD_RMON_EVENTLOG** tag. However, the health monitor sends generic RMON **risingThreshold** and **fallingThreshold** traps.

Are the Ping MIBs returned in decimal notation and ASCII?

Yes, both decimal notation and ASCII are supported, which is the standard implementation in SNMP. All strings are ASCII encoded.

The following example displays the Ping MIB in hexadecimal notation:

```
pingCtlTargetAddress.2.69.72.9.116.99.112.115.97.109.112.108.101 = 0a fa 01 02
```

This translates to ASCII:

```
pingCtlTargetAddress."EH"."tcpsample" = 0a fa 01 02
2= length of the string
69=E
72=H
9=length of second string
116=t
99 =c
112=p
115=s
97=a
109=m
112 =p
108 =l
101 =e
```

As of Junos OS Release 9.6 and later, the Junos OS CLI returns ASCII values using the command **show snmp mib *get* | *get-next* | *walk* ascii**.

The following example shows the output with the ASCII option:

```
user@host> show snmp mib walk pingCtlTargetAddress ascii
pingCtlTargetAddress."EH"."httpgetsample" = http://www.yahoo.com
pingCtlTargetAddress."p1"."t2" = 74 c5 b3 06
pingCtlTargetAddress."p1"."t3" = 74 c5 b2 0c
```

The following example shows the output without the ASCII option:

```
user@host> show snmp mib walk pingCtlTargetAddress
pingCtlTargetAddress.2.69.72.13.104.116.116.112.103.101.116.115.97.109.112.108.101
 = http://www.yahoo.com
pingCtlTargetAddress.2.112.49.2.116.50 = 74 c5 b3 06
pingCtlTargetAddress.2.112.49.2.116.51 = 74 c5 b2 0c
```

You can convert decimal and ASCII values using a decimal ASCII chart like the one at http://www.asciichart.com .

Is IPv6 supported by the Ping MIB for remote operations?

No, IPv6 is not supported.

Is there an SNMP MIB to show Address Resolution Protocol (ARP) table information? Are both IP and MAC addresses displayed in the same table?

Yes, the Junos OS supports the standard MIB **ipNetToMediaTable**, which is described in RFC 2011, *SNMPv2 Management Information Base for the Internet Protocol using SMIv2*. This table is used for mapping IP addresses to their corresponding MAC addresses.

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS SNMP Configuration FAQs on page 361
- SNMPv3 FAQs on page 365
- SNMP Interaction with Juniper Networks Devices FAQs on page 367

Junos OS SNMP Configuration FAQs

This section presents frequently asked questions and answers related to Junos OS SNMP configuration.

Can the Junos OS be configured for SNMPv1 and SNMPv3 simultaneously?

Yes, SNMP has backward compatibility, meaning that all three versions can be enabled simultaneously.

Can I filter specific SNMP queries on a device?

Yes, you can filter specific SNMP queries on a device using **exclude** and **include** statements.

The following example shows a configuration that blocks read-write operation on all OIDs under .1.3.6.1.2.1.1 for the community **test**:

```
user@host# show snmp
view system-exclude {
   oid .1.3.6.1.2.1.1 exclude;
   oid .1 include;
}
   community test {
      view system-exclude;
      authorization read-write;
   }
```

Can I change the SNMP agent engine ID?

Yes, the SNMP agent engine ID can be changed to the MAC address of the device, the IP address of the device, or any other desired value. Several examples are included here.

The following example shows how to use the MAC address of a device as the SNMP agent engine ID:

```
user@host# show snmp
engine-id {
   use-mac-address;
}
```

The following example shows how to use the IP address of a device as the SNMP agent engine ID:

```
user@host# show snmp
engine-id {
   use-default-ip-address;
}
```


The following example shows the use of a selected value, **AA** in this case, as the SNMP agent engine ID of a device:

```
user@host# show snmp
engine-id {
    local AA;
}
```

How can I configure a device with dual Routing Engines or a chassis cluster (for SRX Series Services Gateways or J Series Service Routers) for continued communication during a switchover?

When configuring for continued communication, the SNMP configuration should be identical between the Routing Engines. However, it is best to have separate Routing Engine IDs configured for each Routing Engine, especially when using SNMPv3.

The following example shows the configuration of the Routing Engines in a dual Routing Engine device. Notice that the Routing Engine IDs are set to the MAC addresses for each Routing Engine:

```
user@host# show groups
re0 {
    system {
        host-name PE3-re0;
    }
    interfaces {
        fxp0 {
            unit 0 {
                family inet {
                    address 116.197.178.14/27;
                    address 116.197.178.29/27 {
                        master-only;
                    }
                }
            }
        }
    }
    snmp {
        engine-id {
            use-mac-address;
        }
    }
}
re1 {
    system {
        host-name PE3-re1;
    }
    interfaces {
        fxp0 {
            unit 0 {
                family inet {
                    address 116.197.178.11/27;
                    address 116.197.178.29/27 {
                        master-only;
                    }
                }
```

```
        }
      }
    }
    snmp {
      engine-id {
        use-mac-address;
      }
    }
}
```

The following is an example of an SNMPv3 configuration on a dual Routing Engine device:

```
user@host> show snmp name host1
v3 {
    vacm {
        security-to-group {
            security-model usm {
                security-name test123 {
                    group test1;
                }
                security-name juniper {
                    group test1;
                }
            }
        }
        access {
            group test1 {
                default-context-prefix {
                    security-model any {
                        security-level authentication {
                            read-view all;
                        }
                    }
                }
                context-prefix MGMT_10 {
                    security-model any {
                        security-level authentication {
                            read-view all;
                        }
                    }
                }
            }
        }
    }
    target-address server1 {
        address 116.197.178.20;
        tag-list router1;
        routing-instance MGMT_10;
        target-parameters test;
    }
    target-parameters test {
        parameters {
            message-processing-model v3;
            security-model usm;
            security-level authentication;
            security-name juniper;
```

```
        }
        notify-filter filter1;
    }
    notify server {
        type trap;
        tag router1;
    }
    notify-filter filter1 {
        oid .1 include;
    }
    view all {
        oid .1 include;
    }
    community public {
        view all;
    }
    community comm1;
    community comm2;
    community comm3 {
        view all;
        authorization read-only;
        logical-system LDP-VPLS {
            routing-instance vpls-server1;
        }
    }
    trap-group server1 {
        targets {
            116.197.179.22;
        }
    }
    routing-instance-access;
    traceoptions {
        flag all;
    }
}
```

How can I track SNMP activities?

SNMP trace operations track activity of SNMP agents and record the information in log files.

A sample **traceoptions** configuration might look like this:

```
[edit snmp]
user@host# set traceoptions flag all
```

When the **traceoptions flag all** statement is included at the **[edit snmp]** hierarchy level, the following log files are created:

- snmpd
- mib2d
- rmopd

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS MIBs FAQs on page 353
- SNMPv3 FAQs on page 365
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Traps and Informs FAQs on page 369
- SNMP Support for Routing Instances FAQs on page 376
- SNMP Counters FAQs on page 377

SNMPv3 FAQs

This section presents frequently asked questions and answers related to SNMPv3.

Why is SNMPv3 important?

SNMP v3 provides enhanced security compared to the other versions of SNMP. It provides authentication and encryption of data. Enhanced security is important for managing devices at remote sites from the management stations.

In my system, the MIB object snmpEngineBoots is not in sync between two Routing Engines in a dual Routing Engine device. Is this normal behavior?

Yes, this is the expected behavior. Each Routing Engine runs its own SNMP process (snmpd), allowing each Routing Engine to maintain its own engine boots. However, if both routing engines have the same engine ID and the routing engine with lesser **snmpEngineBoots** value is selected as the master routing engine during the switchover process, the **snmpEngineBoots** value of the master routing engine is synchronized with the **snmpEngineBoots** value of the other routing engine.

Do I need the SNMP manager engine object identifier (OID) for informs?

Yes, the engine OID of the SNMP manager is required for authentication, and informs do not work without it.

I see the configuration of informs under the [edit snmp v3] hierarchy. Does this mean I cannot use informs with SNMPv2c?

Informs can be used with SNMPv2c. The following example shows the basic configuration for SNMPv3 informs on a device (note that the authentication and privacy is set to none):

```
[edit snmp]
v3 {
  usm {
    remote-engine 00000063000100a2c0a845b3 {
      user RU2_v3_sha_none {
        authentication-none;
        privacy-none;
      }
    }
  }
```

```
    vacm {
      security-to-group {
        security-model usm {
          security-name RU2_v3_sha_none {
            group g1_usm_auth;
          }
        }
      }
      access {
        group g1_usm_auth {
          default-context-prefix {
            security-model usm {
              security-level authentication {
                read-view all;
                write-view all;
                notify-view all;
              }
            }
          }
        }
      }
    }
    target-address TA2_v3_sha_none {
      address 192.168.69.179;
      tag-list tl1;
      address-mask 255.255.252.0;
      target-parameters TP2_v3_sha_none;
    }
    target-parameters TP2_v3_sha_none {
      parameters {
        message-processing-model v3;
        security-model usm;
        security-level none;
        security-name RU2_v3_sha_none;
      }
      notify-filter nf1;
    }
    notify N1_all_tl1_informs {
      type inform; # Replace “inform” with “trap” to convert informs to traps.
      tag tl1;
    }
    notify-filter nf1 {
      oid .1 include;
    }
    view all {
      oid .1 include;
    }
  }
```

You can convert the SNMPv3 informs to traps by setting the value of the **type** statement at the **[edit snmp v3 notify N1_all_tl1_informs]** hierarchy level to **trap** as shown in the following example:

```
user@host# set snmp v3 notify N1_all_tl1_informs type trap
```

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Configuration FAQs on page 361
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Traps and Informs FAQs on page 369
- SNMP Support for Routing Instances FAQs on page 376
- SNMP Counters FAQs on page 377

SNMP Interaction with Juniper Networks Devices FAQs

This section presents frequently asked questions and answers related to how SNMP interacts with Juniper Networks devices.

How frequently should a device be polled? What is a good polling rate?

It is difficult to give an absolute number for the rate of SNMP polls per second since the rate depends on the following two factors:

- The number of variable bindings in a protocol data unit (PDU)
- The response time for an interface from the Packet Forwarding Engine

In a normal scenario where no delay is being introduced by the Packet Forwarding Engine and there is one variable per PDU (a Get request), the response time is 130+ responses per second. However, with multiple variables in an SNMP request PDU (30 to 40 for GetBulk requests), the number of responses per second is much less. Because the Packet Forwarding Engine load can vary for each system, there is greater variation in how frequently a device should be polled.

Frequent polling of a large number of counters, especially statistics, can impact the device. We recommend the following optimization on the SNMP managers:

- Use the row-by-row polling method, not the column-by-column method.
- Reduce the number of variable bindings per PDU.
- Increase timeout values in polling and discovery intervals.
- Reduce the incoming packet rate at the SNMP process (snmpd).

For better SNMP response on the device, the Junos OS does the following:

- Filters out duplicate SNMP requests.
- Excludes interfaces that are slow in response from SNMP queries.

One way to determine a rate limit is to note an increase in the **Currently Active** count from the **show snmp statistics extensive** command.

The following is a sample output of the **show snmp statistics extensive** command:

```
user@host> show snmp statistics extensive
  SNMP statistics:
   Input:
     Packets: 226656, Bad versions: 0, Bad community names: 0,
     Bad community uses: 0, ASN parse errors: 0,
     Too bigs: 0, No such names: 0, Bad values: 0,
     Read onlys: 0, General errors: 0,
     Total request varbinds: 1967606, Total set varbinds: 0,
     Get requests: 18478, Get nexts: 75794, Set requests: 0,
     Get responses: 0, Traps: 0,
     Silent drops: 0, Proxy drops: 0, Commit pending drops: 0,
     Throttle drops: 27084, Duplicate request drops: 0
   V3 Input:
     Unknown security models: 0, Invalid messages: 0
     Unknown pdu handlers: 0, Unavailable contexts: 0
     Unknown contexts: 0, Unsupported security levels: 0
     Not in time windows: 0, Unknown user names: 0
     Unknown engine ids: 0, Wrong digests: 0, Decryption errors: 0
   Output:
     Packets: 226537, Too bigs: 0, No such names: 0,
     Bad values: 0, General errors: 0,
     Get requests: 0, Get nexts: 0, Set requests: 0,
     Get responses: 226155, Traps: 382
   SA Control Blocks:
     Total: 222984, Currently Active: 501, Max Active: 501,
     Not found: 0, Timed Out: 0, Max Latency: 25
   SA Registration:
     Registers: 0, Deregisters: 0, Removes: 0
   Trap Queue Stats:
     Current queued: 0, Total queued: 0, Discards: 0, Overflows: 0
   Trap Throttle Stats:
     Current throttled: 0, Throttles needed: 0
   Snmp Set Stats:
     Commit pending failures: 0, Config lock failures: 0
     Rpc failures: 0, Journal write failures: 0
     Mgd connect failures: 0, General commit failures: 0
```

Does SNMP open dynamic UDP ports? Why?

The SNMP process opens two additional ports (sockets): one for IPv4 and one for IPv6. This enables the SNMP process to send traps.

I am unable to perform a MIB walk on the ifIndex. Why is this?

Any variable bindings or values with an access level of **not-accessible** cannot be queried directly because they are part of other variable bindings in the SNMP MIB table. The ifIndex has an access level of **not-accessible**. Therefore, it cannot be accessed directly because it is part of the variable bindings. However, the ifIndex can be accessed indirectly through the variable bindings.

I see SNMP_IPC_READ_ERROR messages when the SNMP process restarts on my system and also during Routing Engine switchover. Is this acceptable?

Yes, it is acceptable to see **SNMP_IPC_READ_ERROR** messages when the SNMP process is restarted, the system reboots, or during a Routing Engine switchover. If all the processes come up successfully and the SNMP operations are working properly, then these messages can be ignored.

What is the source IP address used in the response PDUs for SNMP requests? Can this be configured?

The source IP address used in the response PDUs for SNMP requests is the IP address of the outgoing interface to reach the destination. The source IP address cannot be configured for responses. It can only be configured for traps.

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Configuration FAQs on page 361
- SNMPv3 FAQs on page 365
- SNMP Traps and Informs FAQs on page 369
- SNMP Support for Routing Instances FAQs on page 376
- SNMP Counters FAQs on page 377

SNMP Traps and Informs FAQs

This section presents frequently asked questions and answers related to SNMP traps and informs.

Does the Junos OS impose any rate limiting on SNMP trap generation?

The Junos OS implements a trap-queuing mechanism to limit the number of traps that are generated and sent.

If a trap delivery fails, the trap is added back to the queue, and the delivery attempt counter and the next delivery attempt timer for the queue are reset. Subsequent attempts occur at progressive intervals of 1, 2, 4, and 8 minutes. The maximum delay between the attempts is 8 minutes, and the maximum number of attempts is 10. After 10 unsuccessful attempts, the destination queue and all traps in the queue are deleted.

Junos OS also has a throttle threshold mechanism to control the number of traps sent (default 500 traps) during a particular throttle interval (default 5 seconds). This helps ensure consistency in trap traffic, especially when a large number of traps are generated due to interface status changes.

The throttle interval begins when the first trap arrives at the throttle. All traps within the throttle threshold value are processed, and traps exceeding the threshold value are queued. The maximum size of all trap queues (the throttle queue and the destination queue) is 40,000 traps. The maximum size of any one queue is 20,000 traps. When a trap is added to the throttle queue, or if the throttle queue has exceeded the maximum size, the trap is moved to the top of the destination queue. Further attempts to send the trap from the destination queue are stopped for a 30-second period, after which the destination queue restarts sending the traps.

NOTE: For the Juniper Networks EX Series Ethernet Switch, the maximum size of all trap queues (the throttle queue and the destination queue) is 1,000 traps. The maximum size for any one queue on the EX Series is 500 traps.

I did not see a trap when I had a syslog entry with a critical severity. Is this normal? Can it be changed?

Not every syslog entry with critical severity is a trap. However, you can convert any syslog entry to a trap using the **event-options** statement.

The following example shows how to configure a **jnxSyslogTrap** whenever an **rpd_ldp_nbrdown** syslog entry message error occurs.

```
user@host> show event-options
policy snmptrap {
   events rpd_ldp_nbrdown;
   then {
      raise-trap;
   }
}
```

Are SNMP traps compliant with the Alarm Reporting Function (X.733) on the Junos OS?

No, SNMP traps on the Junos OS are not X.733 compliant.

Can I set up filters for traps or informs?

Traps and informs can be filtered based on the trap category and the object identifier. You can specify categories of traps to receive per host by using the **categories** statement at the **[edit snmp trap-group *trap-group*]** hierarchy level. Use this option when you want to monitor only specific modules of the Junos OS.

The following example shows a sample configuration for receiving only **link**, **vrrp-events**, **services**, and **otn-alarms** traps:

```
[edit snmp]
trap-group jnpr {
   categories {
      link;
      vrrp-events;
      services;
      otn-alarms;
   }
   targets {
      192.168.69.179;
   }
}
```

The Junos OS also has a more advanced filter option (**notify-filter**) for filtering specific traps or a group of traps based on their object identifiers.

The SNMPv3 configuration also supports filtering of SNMPv1 and SNMPv2 traps and excluding Juniper Networks enterprise-specific configuration management traps, as shown in the following configuration example:

```
[edit snmp]
v3 {
    vacm {
        security-to-group {
            security-model v2c {
                security-name sn_v2c_trap {
                    group gr_v2c_trap;
                }
            }
        }
        access {
            group gr_v2c_trap {
                default-context-prefix {
                    security-model v2c {
                        security-level none {
                            read-view all;
                            notify-view all;
                        }
                    }
                }
            }
        }
    }
    target-address TA_v2c_trap {
        address 10.209.196.166;
        port 9001;
        tag-list tg1;
        target-parameters TP_v2c_trap;
    }
    target-parameters TP_v2c_trap {
        parameters {
            message-processing-model v2c;
            security-model v2c;
            security-level none;
            security-name sn_v2c_trap;
        }
        notify-filter nf1;
    }
    notify v2c_notify {
        type trap;
        tag tg1;
    }
    notify-filter nf1 {
        oid .1.3.6.1.4.1.2636.4.5 exclude;
        oid .1 include;
    }
    snmp-community index1 {
        community-name "$9$tDLl01h7Nbw2axN"; ## SECRET-DATA
        security-name sn_v2c_trap;
        tag tg1;
    }
view all {
```

```
    oid .1 include;
  }
  }
```

Can I simulate traps on a device?

Yes, you can use the **request snmp spoof-trap *trap name*** command for simulating a trap to the NMS that normally receives your device's traps. You can also add required values using the **variable-bindings** parameter.

The following example shows how to simulate a trap to the local NMS using variable bindings:

```
user@host> request snmp spoof-trap linkDown variable-bindings "ifIndex[116]=116,
  ifAdminStatus[116]=1 ,ifOperStatus[116]=2 , ifName[116]=ge-1/0/1"
```

How do I generate a warm start SNMPv1 trap?

When the SNMP process is restarted under normal conditions, a warm start trap is generated if the system up time is more than 5 minutes. If the system up time is less than 5 minutes, a cold start trap is generated.

The NMS sees only the MIB OIDs and numbers, but not the names of the SNMP traps. Why?

Before the NMS can recognize the SNMP trap details, such as the names of the traps, it must first compile and understand the MIBs and then parse the MIB OIDs.

In the Junos OS, how can I determine to which category a trap belongs?

For a list of common traps and their categories, see *Juniper Networks Enterprise-Specific SNMP Version 1 Traps* and *Juniper Networks Enterprise-Specific SNMP Version 2 Traps* in the *Junos OS SNMP MIBs and Traps Reference* document.

Can I configure a trap to include the source IP address?

Yes, you can configure the **source-address**, **routing-instance**, or **logical-instance** name for the source IP address using the **trap-options** command:

```
user@host> show snmp trap-options
source-address 10.1.1.1;
```

Can I create a custom trap?

Yes, you can use the **jnxEventTrap** event script to create customized traps as needed.

In the following example, a Junos OS operations (op) script is triggered when a **UI_COMMIT_NOT_CONFIRMED** event is received. The Junos OS op script matches the complete message of the event and generates an SNMP trap.

Example: Junos OS Op Script

```
version 1.0;

ns junos = "http://xml.juniper.net/junos/*/junos";
ns xnm = "http://xml.juniper.net/xnm/1.1/xnm";
ns jcs = "http://xml.juniper.net/junos/commit-scripts/1.0";

param $event;
param $message;

match / {

    /*
     * trapm utilty wants the following characters in the value to be escaped
     * '[', ']', ' ', '=', and ','
     */
    var $event-escaped = {
        call escape-string($text = $event, $vec = '[] =,');
    }

    var $message-escaped = {
        call escape-string($text = $message, $vec = '[] =,');
    }

    <op-script-results> {
 var $rpc = <request-snmp-generate-trap> {
     <trap> "jnxEventTrap";
     <variable-bindings> "jnxEventTrapDescr[0]='Event-Trap' , "
  _ "jnxEventAvAttribute[1]='event' , "
  _ "jnxEventAvValue[1]='" _ $event-escaped _ "' , "
  _ "jnxEventAvAttribute[2]='message' , "
  _ "jnxEventAvValue[1]='" _ $message-escaped _ "'";
 }

 var $res = jcs:invoke($rpc);
    }
}

template escape-string ($text, $vec) {

    if (jcs:empty($vec)) {
        expr $text;

    } else {
        var $index = 1;
        var $from = substring($vec, $index, 1);
        var $changed-value = {
            call replace-string($text, $from) {
                with $to = {
                    expr "\\";
                    expr $from;
                 }
            }
        }

        call escape-string($text = $changed-value, $vec = substring($vec, $index

 + 1));
    }
}

template replace-string ($text, $from, $to) {
```

```
    if (contains($text, $from)) {
        var $before = substring-before($text, $from);
        var $after = substring-after($text, $from);
        var $prefix = $before _  $to;

        expr $before;
        expr $to;
        call replace-string($text = $after, $from, $to);

    } else {
        expr $text;
    }
}
```

After creating your customized trap, you must configure a policy on your device to tell the device what actions to take after it receives the trap.

Here is an example of a configured policy under the **[edit event-options]** hierarchy:

```
[edit event-options]
user@host> show
policy trap-on-event {
  events UI_COMMIT_NOT_CONFIRMED;
  attributes-match {
    UI_COMMIT_NOT_CONFIRMED.message matches complete;
  }
  then {
    event-script ev-syslog-trap.junos-op {
      arguments {
        event UI_COMMIT_NOT_CONFIRMED;
        message "{$$.message}";
      }
    }
  }
}
```

Can I disable link up and link down traps on interfaces?

Yes, link up and link down traps can be disabled in the interface configuration. To disable the traps, use the **no-traps** statement at the **[edit interfaces *interface-name* unit *logical-unit-number*]** and **[edit logical-systems *logical-system-name* interfaces *interface-name* unit *logical-unit-number*]** hierarchies for physical and logical interfaces.

```
(traps | no-traps);
```

I see the link up traps on logical interfaces, but I do not see the link down traps. Is this normal behavior?

For Ethernet and ATM types of interfaces, Junos OS does not send link down traps for a logical interface if the physical interface is down to prevent flooding alarms for the same root cause. However, when the physical interface and logical interfaces come back up, traps are sent indicating link up. This is because the physical interface coming up does not necessarily mean the logical interfaces are also coming up.

For SONET types of interfaces with PPP encapsulation, Junos OS does send link down traps for a logical interface if the physical interface is down. When the physical interface and logical interfaces come back up, traps are sent for both the physical and logical interfaces indicating link up.

For SONET types of interfaces with HDLC encapsulation, Junos OS does not send link down traps for a logical interface if the physical interface is down. When the physical interface and logical interfaces come back up, traps are sent for both the physical and logical interfaces indicating link up.

For channelize interfaces with PPP encapsulation, Junos OS does send link down traps for a logical interface if the physical interface is down. When the physical interface and logical interfaces come back up, traps are sent for both the physical and logical interfaces indicating link up.

For channelize interfaces with HDLC encapsulation, Junos OS does not send link down traps for a logical interface if the physical interface is down. When the physical interface and logical interfaces come back up, traps are sent for both the physical and logical interfaces indicating link up.

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Configuration FAQs on page 361
- SNMPv3 FAQs on page 365
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Support for Routing Instances FAQs on page 376
- SNMP Counters FAQs on page 377

Junos OS Dual Routing Engine Configuration FAQs

This section presents frequently asked questions and answers related to the configuration of dual Routing Engines.

The SNMP configuration should be identical between the Routing Engines when configuring for continued communication. However, we recommend having separate Routing Engine IDs configured for each Routing Engine, when using SNMPv3.

In my system, the MIB object snmpEngineBoots is not in sync between two Routing Engines in a dual Routing Engine device. Is this normal behavior?

Yes. This is the normal behavior. Each Routing Engine runs its own SNMP process (snmpd) agent, allowing each Routing Engine to maintain its own engine boots.

Is there a way to identify that an address belongs to RE0, RE1, or the master Routing Engine management interface (fxp0) by looking at an SNMP walk?

No. When you do an SNMP walk on the device, it only displays the master Routing Engine management interface address.

What is the best way to tell if the current IP address belongs to fxp0 or a Routing Engine, from a CLI session?

Routing Engines are mapped with the **fxp0** interface. This means that when you query RE0, the ifTable reports the **fxp0** interface address of RE0 only. Similarly, if you query RE1, the ifTable reports the **fxp0** interface address of RE1 only.

When there is a failover, the master hostname is changed since the hostname belongs to the Routing Engine. Is this correct?

Yes. You can configure the same hostname or different hostnames. Either would work.

If only the master IP address is configured (for example, 192.168.2.5), and the **sysDescr.0** object has the same string configured on both of the Routing Engines, then even after a switchover, the **sysDescr.0** object returns the same value. The following sample shows the results you get by using the **snmpget** command:

```
bng-junos-pool02: /c/svivek/PR_BRANCH/src> snmpget -c jnpr -v2c 192.168.2.5
sysDescr.0 system.sysDescr.0 = foo
```

SNMP Support for Routing Instances FAQs

This section presents frequently asked questions and answers related to how SNMP supports routing instances.

Can the SNMP manager access data for routing instances?

Yes, the Junos OS enables SNMP managers for all routing instances to request and manage SNMP data related to the corresponding routing instances and logical system networks.

Two different routing instance behaviors can occur, depending on where the clients originate:

- Clients from routing instances other than the default can access MIB objects and perform SNMP operations only on the logical system networks to which they belong.
- Clients from the default routing instance can access information related to all routing instances and logical system networks.

Routing instances are identified by either the context field in SNMPv3 requests or encoded in the community string in SNMPv1 or SNMPv2c requests.

When encoded in a community string, the routing instance name appears first and is separated from the actual community string by the @ character.

To avoid conflicts with valid community strings that contain the @ character, the community is parsed only if typical community string processing fails. For example, if a routing instance named **RI** is configured, an SNMP request with **RI@public** is processed within the context of the **RI** routing instance. Access control (including views, source address restrictions, and access privileges) is applied according to the actual community string (the set of data after the @ character—in this case **public**). However, if the

community string **RI@public** is configured, the PDU is processed according to that community, and the embedded routing instance name is ignored.

Logical systems perform a subset of the actions of a physical router and have their own unique routing tables, interfaces, policies, and routing instances. When a routing instance is defined within a logical system, the logical system name must be encoded along with the routing instance using a slash (/) to separate the two. For example, if the routing instance **RI** is configured within the logical system **LS**, that routing instance must be encoded within a community string as **LS/RI@public**. When a routing instance is configured outside a logical system (within the default logical system), no logical system name, or **/** character, is needed.

Additionally, when a logical system is created, a default routing instance named **default** is always created within the logical system. This name should be used when querying data for that routing instance, for example **LS/default@public**. For SNMPv3 requests, the name ***logical system/routing instance*** should be identified directly in the context field.

Can I access a list of all routing instances on a device?

Yes, you can access a list of all the routing instances on a device using the vacmContextName object in the SNMP-VIEW-BASED-ACM MIB. In SNMP, each routing instance becomes a VACM context; this is why the routing instances appear in the vacmContextName object.

Can I access a default routing instance from a client in another logical router or routing instance?

No, the SNMP agent can only access data of the logical router to which it is connected.

Related Documentation

- Junos OS SNMP Support FAQs on page 352
- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Configuration FAQs on page 361
- SNMPv3 FAQs on page 365
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Traps and Informs FAQs on page 369
- SNMP Counters FAQs on page 377

SNMP Counters FAQs

This section presents frequently asked questions and answers related to SNMP counters.

Which MIB should I use for interface counters?

Interface management over SNMP is based on two tables: the **ifTable** and its extension the **ifXTable**. Both are described in RFC 1213, *Management Information Base for Network Management of TCP/IP-based internets: MIB-II* and RFC 2233, *The Interfaces Group MIB using SMIv2*.

Interfaces can have several layers, depending on the media, and each sublayer is represented by a separate row in the table. The relationship between the higher layer and lower layers is described in the **ifStackTable**.

The **ifTable** defines 32-bit counters for inbound and outbound octets (ifInOctets/ifOutOctets), packets (ifInUcastPkts/ifOutUcastPkts, ifInNUcastPkts /ifOutNUcastPkts), errors, and discards.

The **ifXTable** provides similar 64-bit counters, also called high capacity (HC) counters, for inbound and outbound octets (ifHCInOctets/ifHCOutOctets) and inbound packets (ifHCInUcastPkts).

When should 64-bit counters be used?

It is always good to use 64-bit counters because they contain statistics for both low and high capacity components.

Are the SNMP counters ifInOctets and ifOutOctets the same as the command reference show interfaces statistics in and out counters?

Yes, these are the same, but only if SNMP is enabled when the router boots up. If you power on a Juniper Networks device and then enable SNMP, the SNMP counters start from 0. SNMP counters do not automatically receive their statistics from the **show** command output. Similarly, using the **clear statistics** command does not clear the statistics that the SNMP counters collected, which can cause a discrepancy in the data that is seen by both processes.

Do the SNMP counters ifInOctets and ifOutOctets include the framing overhead for Point-to-Point Protocol (PPP) and High-Level Data Link Control (HDLC)?

Yes.

Related Documentation

- Junos OS SNMP FAQs Overview on page 351
- Junos OS MIBs FAQs on page 353
- Junos OS SNMP Support FAQs on page 352
- SNMPv3 FAQs on page 365
- Junos OS SNMP Configuration FAQs on page 361
- SNMP Interaction with Juniper Networks Devices FAQs on page 367
- SNMP Traps and Informs FAQs on page 369
- SNMP Support for Routing Instances FAQs on page 376

PART 5

Index

Index

Symbols

A

B

C

D

E

F

G

N

O

P

R

S

T

U

V

W