Chapter 4
# Configuring SNMP

This chapter provides information for configuring Simple Network Management Protocol (SNMP) on your E-series router.

This chapter contains the following sections:



## Overview

SNMP is a protocol that manages network devices, such as your E-series router. The goal of SNMP is to simplify network management in two ways:

- By defining a single management protocol that can be used to manage any network device from any vendor.

  This feature reduces the complexity of the network management application because the application does not need to support a large number of proprietary management protocols for the mix of vendors' devices in the network.

- By defining a single, consistent representation of managed information that is commonly deployed in network devices.

  For example, SNMP uses a common form and semantics for interface statistics, a process that supports consistent interpretation and meaningful comparison.

SNMP is an application-level protocol that comprises the following three elements:

- An SNMP client (manager)
- An SNMP server (agent)
- A Management Information Base (MIB)

SNMP defines a client-server model in which a client (*manager*) obtains information from the server (*agent*) through two mechanisms:

- A request/response protocol by which the client configures and monitors the server. In this instance, the information is solicited.
- Asynchronous notifications (called *traps*) by which the server, on its own initiative, reports notable changes in the router's status to the client. In this instance, the information is unsolicited.

## Terminology

Table 17 provides definitions for the basic SNMP terms.

**Table 17: SNMP Terminology**

| Term | Meaning |
|---|---|
| agent | Also referred to as server; a managed device, such as a router, that collects and stores management information |
| client | Sometimes called a network management station (NMS) or simply a manager; a device that executes management applications that monitor and control network elements |
| community | A logical group of SNMP-managed devices and clients in the same administrative domain |
| entity | Refers to both a server and a client |
| event | A condition or state change that may cause the generation of a trap message |
| managed object | A characteristic of something that can be managed, such as a list of currently active TCP circuits in a device |
| group | SNMPv3 term; a set of users with the same access privileges to the router |
| MIB | Management Information Base; a collection of managed objects residing in a virtual information store |
| network element | Also known as a managed device; a hardware device, such as a PC or a router |
| notification | A message that indicates a status change (equivalent to a trap) |
| server | Also referred to as agent; a managed device, such as a router, that collects and stores management information |

**Table 17: SNMP Terminology (continued)**

| Term | Meaning |
|---|---|
| trap | Message sent by an SNMP server to a client to indicate the occurrence of a significant event, such as a specifically defined condition or a threshold that was reached. Managed devices use traps to asynchronously report certain events to clients. |
| user | SNMPv3 term; an individual who accesses the router |
| view | SNMPv3 term; defines the management information available to the user: read, write, or notification |

## SNMP Features Supported

This SNMP implementation provides the following:

- Standard SNMP MIB support for services and interfaces as defined by the Internet Engineering Task Force (IETF)
- A set of AS number version 1 notated enterprise MIBs for all management functions not addressed by standard MIBs
- A multilingual SNMP server that supports SNMPv1, SNMPv2c, and SNMPv3 protocols
- Enhanced security and management features supported in SNMPv3
- Traps for alarm and state change events
- Bulk data collection and retrieval
- Management of virtual routers
- Secure audit logging for packet mirroring traps and Juni-PACKET-MIRROR MIB access

**NOTE:** You can disable the management interface through SNMP. But, if you disable the management interface, you can no longer access the router through SNMP.

**NOTE:** JUNOSe software supports SNMP packet mirroring traps; however, the packet mirroring-related SNMP commands, categories, and traps are visible in the CLI only to authorized users. See *JUNOSe Policy Management Configuration Guide, Chapter 10, Packet Mirroring Overview* for information about using SNMP with secure packet mirroring.

## SNMP Client

The SNMP client runs on a network host and communicates with one or more SNMP servers on other network devices, such as routers, to configure and monitor the operation of those network devices.

## SNMP Server

The SNMP server operates on a network device, such as a router, a switch, or a workstation. It responds to SNMP requests received from SNMP clients and generates *trap messages* to alert the client(s) about notable state changes in the network device.

The SNMP server implementation operates over UDP/IP only. It can receive requests directed to any IP address configured on the router. SNMP requests and responses are received or sent on UDP port 161. SNMP traps are sent from UDP port 162 by default or from a configurable port. For traps sent from UDP port 162, you can configure the destination UDP port for each recipient with the **snmp-server host** command.

## SNMP MIBs

A MIB specifies the format of managed data for a device function. The goal of a MIB is to provide a common and consistent management representation for that function across networking devices.

Your router supports both standard and enterprise SNMP MIBs.

### Standard SNMP MIBs

A standard MIB is defined by a body such as the IETF and fosters consistency of management data representation across many vendors' networking products.

### Juniper Networks E-series Enterprise MIBs

An enterprise MIB is defined by a single vendor. In addition to providing consistency of management data representation across that vendor's product line, the enterprise MIB also accounts for proprietary functions and value-added features not addressed by standard MIBs.

For example, boot record extensions to the enterprise MIB enable configuration of the release (.rel) files for each router, slot, and subsystem. The extensions also enable booting through the factory defaults, the running configuration, or a configuration (.cnf) file.

### Accessing Supported SNMP MIBs

For complete information about the SNMP MIBs supported by your router, see the *E-series System Software* CD, shipped with your router. In the MIBs folder you will find information about all supported standard and Juniper Networks E-series Enterprise (proprietary) MIBs.

## SNMP Versions

This SNMP server implementation supports:

- SNMPv1 (defined in RFC 1157)
- SNMPv2c (Community-based SNMPv2, defined in RFC 1901 and RFC 3416)
- SNMPv3 (compliant with RFCs 3410–3418, STD 62)

The server encodes SNMP responses using the same SNMP version received in the corresponding request and encodes traps using the SNMP version configured for the trap recipient.

SNMPv2c supports the capabilities defined for SNMPv1 and provides greater power and flexibility through the addition of several features, including:

- More detailed error codes
- GetBulk operation for efficient retrieval of large amounts of data
- 64-bit counters

SNMPv3 is an extensible SNMP framework that supplements the SNMPv2c framework by supporting:

- Security for messages
- Explicit access control

### *Security Features*

As users transfer more sensitive information, such as billing details, through the Internet, security becomes more critical for SNMP and other protocols. SNMPv3 provides the user-based security model (USM) to address authentication and data encryption.

Authentication provides the following benefits:

- Only authorized parties can communicate with each other. Consequently, a management station can interact with a device only if the administrator configured the device to allow the interaction.
- Messages are received promptly; users cannot save messages and replay them to alter content. This feature prevents users from sabotaging SNMP configurations and operations. For example, users can change configurations of network devices only if authorized to do so.

SNMPv3 authenticates users through the HMAC-MD5-96 or HMAC-SHA-96 protocols; CBC-DES is the encryption or privacy protocol. The SNMP agent recognizes up to 32 usernames that can have one of the following security levels:

- No authentication and no privacy (none)
- Authentication only (auth only)
- Authentication and privacy (priv)

In contrast, SNMPv1and SNMPv2c provide only password protection, through the community name and IP address. When an SNMP server receives a request, the server extracts the client's IP address and the community name. The SNMP community table is searched for a matching community. If a match is found, its access list, if nonzero, is used to validate the IP address. If the access list number is zero, the IP address is accepted. A nonmatching community or an invalid IP address causes an SNMP authentication error. Each entry in the community table identifies:

- An SNMP community name
- An SNMP view name
- A user's privilege level
  - Read-only (ro)
  - Read-write (rw)
  - Administrator (admin)
- An IP access list name

### *Management Features*

Management features of SNMPv3 allow you to specify who will receive notifications and to define MIB views that users in different groups can access:

- Notification—Message that informs you of a status change; the equivalent of a trap in SNMPv1.
- View—Definition of the management information that is available: read, write, or notification. Predefined views are available for each group:
  - everything—Includes all MIBs associated with the router, except the packetMirror MIB
  - user—Includes all MIBs associated with the router, except the packetMirror MIB and standard and enterprise MIBs used to configure SNMP operation
  - nothing—Excludes all MIBs
  - mirrorAdmin—Includes the packetMirror MIB
- User—An individual who requires access to the router. The router may provide authentication and privacy for the user through SNMPv3. Each user is associated with a group.
- Group—A set of users with the same access privileges to the router. Three predefined groups are available: admin, public, and private. Table 18 shows the security levels and views associated with these groups.

**Table 18: Relationship Among Groups, Security Levels, and Views**

| Group Name | Security Level | Read View | Write View | Notification/ Trap View |
|---|---|---|---|---|
| admin | authentication and privacy | everything | everything | everything |
| mirror | authentication and privacy | mirrorAdmin | mirrorAdmin | mirrorAdmin |
| public | none | user | nothing | nothing |
| private | authentication only | user | user | user |

## *Virtual Routers*

All SNMP-related CLI commands operate in the context of a virtual router, which means that you must configure users, traps, communities, and so on for *each* server. You must set the context using the **virtual-router** command and then configure SNMP.

The **show snmp** commands show only statistics and configuration information for the server/SNMP agent that corresponds to the current virtual router context.

The exceptions to this convention are the **snmp-server contact** and the **snmp-server location** commands. With these commands, single instances of the contact and the location are created regardless of the number of virtual routers.

### Creating SNMP Proxy

Your JUNOSe software allows you to configure multiple virtual routers. Each virtual router has its own SNMP server. At router initialization, SNMP creates a server for each existing virtual router.

When router-specific data is required, the requestor can direct a request to a particular server for a virtual router through the base community string extension: for example, SNMP get public@megaRouter.

**NOTE:** In addition to the @ selector character, the system also supports the % selector character. For example, SNMP get public%megaRouter.

When any system server parses a request and detects an extended community string, it acts as proxy by forwarding the request to the server corresponding to the virtual router name in the extension (for example, *megaRouter)*. The target server then processes the request and generates a response, which is then returned to the proxy server and subsequently transmitted to the requester.

The JUNOSe implementation of SNMPv3 communicates with virtual routers by assigning each proxy agent an SNMP engine ID. This difference is unimportant to users of the CLI. However, if you use other SNMPv3 applications to manage the router, refer to the following section.

### Disabling and Reenabling SNMP Proxy

The ability to proxy SNMP from a virtual router (VR) is enabled by default whenever you create a virtual router agent. However, you can disable or reenable the proxy feature on each virtual router agent to address any network security issues. To disable proxy on an agent (router), you must use SNMP or the CLI **snmp-server proxy disable** command.

**NOTE:** Disabling the proxy function on a particular virtual router disables the use of proxy through that virtual router. You can, however, use the proxy function to access a proxy-disabled virtual router through another virtual router that does have the proxy function enabled.

## *Communicating with the SNMP Engine*

The SNMP engine performs the following tasks for SNMPv3:

- Sends and receives messages.
- Prepares messages and extracts data from messages.
- Authenticates, encrypts, and decrypts messages.
- Determines whether access to a managed task is allowed.

Each SNMP engine has an SnmpEngine ID, a hexadecimal number 15 octets long. Table 19 shows the structure of the SnmpEngine ID.

**Table 19: SnmpEngineID Structure Object**

| Octet Assignment | Description |
|---|---|
| 1 – 4 | E-series router SNMP management private enterprise number |
| 5 | Indicates that octets 6–15 contain information determined by the E-series router |
| 6 – 11 | The MAC address for the device |
| 12 – 15 | The 32-bit (4 octet) router index (or routerUID) |

Request protocol data units (PDUs) for the SNMP engine must contain the corresponding contextEngine ID and contextName for the SNMP engine. When the system receives a PDU, it examines the contextEngine ID and contextName, and forwards the request to the corresponding virtual router.

- The contextEngine ID is the same as the SnmpEngine ID.
- The contextName is an internally derived ASCII string associated with the router. It has the format *routerN*, where N is a number (with no leading zeros) in the range 1–16777215, corresponding to the least significant 24 bits of the 32-bit router index (or router UID). You can obtain the contextName for a specific router through the Juniper-ROUTER-MIB from the juniRouterContextName object in the juniRouterTable, which is indexed by the 32-bit router index (juniRouterIndex).

The following table shows examples of the E-series router SNMP engine objects that are associated with the default virtual router.

| Object | Value |
|---|---|
| SnmpEngineID | 0x80:00:13:0a:05:00:90:1a:00:04:6c:80:00:00:01 |
| contextEngineID | 0x80:00:13:0a:05:00:90:1a:00:04:6c:80:00:00:01 |
| contextName | router1 |

## SNMP Attributes

The software automatically maps predefined SNMPv1/v2c attributes to predefined SNMPv3 attributes, as shown in Table 20.

**Table 20: Relationship Between SNMPv1/v2c and SNMPv3 Attributes**

| Attribute | SNMPv1/v2C Value | SNMPv3 Value |
|---|---|---|
| Community | admin | admin |
| View | | everything |
| Privilege | rw | rw |
| | | |
| Community | public | public |
| View | | user |
| Privilege | ro | ro |
| | | |
| Community | private | private |
| View | | user |
| Privilege | rw | rw |

## SNMP Operations

SNMP has the five operations defined in Table 21.

**Table 21: SNMP Operations**

| SNMP Operation | Definition |
|---|---|
| Get | Allows the client to retrieve an object instance from the server. |
| GetNext | Allows the client to retrieve the next object instance from a table or list within a server. |
| GetBulk | Makes it easier to acquire large amounts of related information without initiating repeated GetNext operations. GetBulk is not available in SNMPv1. |
| Set | Allows the client to set values for the objects managed by the server. |
| Notification | Used by the server to asynchronously inform the client of some event. (Called a trap in SNMPv1.) |

### *SNMP PDU Types*

SNMP offers the six types of PDUs defined in Table 22.

**Table 22: SNMP PDU Types**

| SNMP PDU Type | Definition |
|---|---|
| Get Bulk | Transmitted by the client to the server to obtain the identifiers and the values of a group or collection of variables rather than one variable at a time. GetBulk is not available in SNMPv1. |
| Get Next Request | Transmitted by the client to the server to obtain the identifiers and the values of variables located *after* the designated variables. |
| Get Request | Transmitted by the client to the server to obtain the values of designated variables. |
| Get Response | Transmitted by the server to the client in response to a Get Request, a Get Next Request, or a Set Request PDU. |
| Set Request | Transmitted by the client to the server to modify the values of designated variables. |
| Notification | Transmitted by the server, on its own initiative, to inform the client of a special event noted on a network device. (Called a trap in SNMPv1.) |

## Platform Considerations

SNMP is supported on all E-series routers.

For information about the modules supported on E-series routers:

- See the *ERX Module Guide* for modules supported on ERX-7xx models, ERX-14xx models, and the ERX-310 router.
- See the *E120 and E320 Module Guide* for modules supported on the E120 router and the E320 router.

## References

For more information about SNMP, consult the following resources:

- RFC 1157—A Simple Network Management Protocol (SNMP) (May 1990)
- RFC 1901—Introduction to Community-based SNMPv2 (January 1996)
- RFC 2790—Host Resources MIB (March 2000)
- RFC 2864—The Inverted Stack Table Extension to the Interfaces Group MIB (June 2000)
- RFC 2493—Textual Conventions for MIB Modules Using Performance History Based on 15 Minute Intervals (January 1999)
- RFC 3014—Notification Log MIB (November 2000)

- RFC 3410—Introduction and Applicability Statements for Internet Standard Management Framework (December 2002)
- RFC 3411—An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks (December 2002)
- RFC 3412—Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) (December 2002)
- RFC 3413—Simple Network Management Protocol (SNMP) Applications (December 2002)
- RFC 3414—User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3) (December 2002)
- RFC 3415—View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP) (December 2002)
- RFC 3416—Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP) (December 2002)

## Before You Configure SNMP

Before you configure SNMP, make sure that at least one IP address is configured on your router. See *ERX Hardware Guide, Chapter 7, Accessing ERX Routers* or *E120 and E320 Hardware Guide, Chapter 7, Accessing E-series Routers*.

Also make sure that you have the necessary configuration information for:

- Communities and their assigned privileges
- IP addresses of SNMP clients and trap recipients
- SNMPv3 users

## SNMP Configuration Tasks

To configure the SNMP server:

1. Enable the SNMP server.

   host1(config)#**snmp-server**

2. Configure at least one authorized SNMP community (SNMPv1/v2c) or user (SNMPv3), which provides SNMP client access.

   host1(config)#**snmp-server community boston view everything rw**
   host1(config)#**snmp-server user fred group private auth sha fred-password priv des password**

3. (Optional) Set the server parameters—contact name and server location.

   host1(config)#**snmp-server contact Bob Smith**
   host1(config)#**snmp-server location 3rdfloor**

4. (Optional) Reconfigure the maximum SNMP packet size.

   host1(config)#**snmp-server packetsize 1000**

5. (Optional) Configure memory warning parameters.

   host1(config)#**memory warning 80 70**

6. (Optional) Configure the method the router uses to encode the ifDescr and ifName objects.

   host1(config)#**snmp interfaces description-format common**

7. (Optional) Manage the interface sublayers (compress interfaces and control interface numbering).

   host1(config)#**snmp-server interfaces compress atmAal5**
   host1(config)#**snmp-server interface compress-restriction ifadminstatusdown**
   host1(config)#**snmp interfaces rfc1213 55000 100000**

8. (Optional) Configure the dynamic group parameters.

   host1(config)#**snmp-server group grp1authpriv usm priv read grp1read write grp1write notify grp1notify**

9. (Optional) Configure the dynamic view parameters.

   host1(config)#**snmp-server view view1 1.3.6.1 included non-volatile**

You can also set up SNMP traps and set up the router to collect bulk statistics. See *Configuring Traps* on page 153 and *Collecting Bulk Statistics* on page 178.

## Enabling SNMP

To enable the SNMP server, use the following command.

### *snmp-server*

- Use to enable SNMP server operation.
- Example

  host1(config)#**snmp-server**

- Use the **no** version to disable the SNMP server operation.

## *Configuring SNMP v1/v2c Community*

For SNMPv1/v2c, access to an SNMP server by an SNMP client is governed by a proprietary SNMP community table that identifies those communities that have read-only, read-write, or administrative permission to the SNMP MIB stored on a particular server.

When an SNMP server receives a request, the server extracts the client's IP address and the community name. The SNMP community table is searched for a matching community. If a match is found, its access list name is used to validate the IP address. If the access list name is null, the IP address is accepted. A nonmatching community or an invalid IP address results in an SNMP authentication error.

Each entry in the community table identifies:

- An SNMP community name
- A user's privilege level
- An IP access list

### Community Name

The community name acts as a password and is used to authenticate messages sent between an SNMP client and a router containing an SNMP server. The community name is sent in every packet between the client and the server.

### Privilege Levels

SNMP has three privilege levels:

- Read-only—Read-only access to the entire MIB except for SNMP configuration objects
- Read-write—Read-write access to the entire MIB except for SNMP configuration objects
- Admin—Read-write access to the entire MIB

### IP Access List

The IP access list identifies those IP addresses of SNMP clients permitted to use a given SNMP community.

***snmp-server community***

- Use to configure an authorized SNMP community for access to the SNMP MIBs and to associate SNMPv1/v2c communities with SNMP MIB views.
- The community name serves as a password and permits access to an SNMP server. The name can be up to 31 characters, and it must be enclosed in quotation marks.
- The maximum number of communities in each virtual router is 32.
- By default, an SNMP community permits only read-only access.
- The view name allows configuration with available dynamic views.

- Example

  host1(config)#**snmp-server community “boston” view view1 rw**

- Use the **no** version to delete a community from the SNMP community table.

## Configuring SNMPv3 Users

To configure SNMPv3 users, use the following command.

***snmp-server user***

- Use to create and modify SNMPv3 users.
- Example

  host1(config)#**snmp-server user fred auth sha fred-password priv des password group user**

- Use the **no** version to delete users.

## Configuring SNMP Dynamic Groups and Views

With dynamic configurable views and groups you can fine-tune application features to a specific group, You can have 32 view entries (with distinct names) per virtual router. Because there is no limit to the number of entries within a distinct view name, you can configure complex views. You can also have 32 access entries (with distinct names) per virtual router. All views are on a per-virtual-router basis; although static views are on a per-virtual-router basis, they cannot be altered. If you modify a view, the system deletes the original entry and creates the new view. Therefore, if the new view fails, the original view is no longer available.

SNMP v3 configurations are allowed only at the maximum CLI privilege level (15).

***snmp-server group***

- Use to dynamically configure server groups. You must access the CLI at privilege level 15 to view or use this command.
- Example

  host1(config-profile)#**snmp-server group grp1authpriv usm priv read grp1read write grp1write notify grp1notify**

- Use the **no** version to remove the dynamically created group.

***snmp-server view***

- Use to dynamically configure an SNMP server view. You must access the CLI at privilege level 15 to view or use this command.
- Example

  host1(config)#**snmp-server view view1 1.3.6.1 included non-volatile**

- Use the **no** version to remove the dynamically created view.

## Setting Server Parameters

Setting the server's contact person and location provides helpful identifiers for the SNMP server. These identifiers are arbitrary and do not affect the server's function, but they are useful to have.

***snmp-server contact***
***snmp-server location***

- Use these commands to configure the SNMP server's contact person and the server's location.
- The contact is the person who manages the server.
- The location is the server's physical location.
- Each of these parameters can be up to 64 characters.
- Example

  host1(config)#**snmp-server contact Bob Smith**
  host1(config)#**snmp-server location 3rdfloor**

- Use the **no** version of these commands to clear the contact or location identifier from the SNMP configuration.

## Configuring SNMP Packet Size

The SNMP server must support a PDU with an upper limit of 484 bytes or greater. There is no need to coordinate the maximum packet size across the entire network. Many requests and responses tend to be smaller than the maximum value.

***snmp-server packetsize***

- Use to set the SNMP server's maximum packet size.
- Increase this value to improve the efficiency of the GetBulk operation.
- Example

  host1(config)#**snmp-server packetsize 1000**

- Use the **no** version to set the SNMP packet size to the default maximum size, 1500 bytes.

## Configuring Memory Warning

You can set up the router to send memory warning messages when memory utilization reaches a specified value.

***memory***

- Use to configure memory warning parameters. You set a high memory utilization value and an abated memory utilization value. When the system reaches the high utilization value, it sends warning messages. When memory usage falls to the abated utilization value, the system stops sending warning messages.

- Example

  host1(config)#**memory warning 80 70**

- Use the **no** version to return to the default values, 85 for high utilization and 75 for abated memory utilization.

## *Configuring Encoding Method*

You can control how the router encodes the ifDescr and ifName objects in the SNMP agent's interface table and in the bulkstats application.

There are two choices of encoding schemes: an E-series router proprietary method and a conventional industry method.

- The proprietary method identifies each interface sublayer with its type.
- The industry method bases the type information for each interface sublayer on the lowest layer 1 or layer 2 interface type.

For example a PPP interface configured on top of an ATM interfaces is:

- PPP3/0.1—Proprietary method
- ATM3/0.1—Industry method

### *snmp-server interfaces description-format*

- Use to set the encoding scheme of the ifDescr and ifName objects. Include one of the following keywords:
  - **common**—Sets the encoding scheme to the conventional industry method and provides compatibility with software that uses the industry encoding scheme.
  - **legacy**—Sets the encoding scheme for legacy E-series routers.
  - **proprietary—**Sets the encoding scheme to the E-series router proprietary method.
- Example

  host1(config)#**snmp-server interfaces description-format common**

- Use the **no** version to return to the default, the legacy encoding scheme.

## *Managing Interface Sublayers*

You can set up the SNMP agent to compress the number of interface instances in the standard interface and stack tables. You can also control the interface numbering method used in the interface tables.

### Compressing Interfaces

You can compress interfaces by interface type and by the administrative status of the interface. Compressing interfaces removes them from the ifTable, the ifStackTable, and the ipAddrTable, which increases table retrieval performance. For example, if you want statistics kept only on IP interfaces, then you can compress all interfaces except IP; subsequently, only IP interfaces will appear in the ifTable, the ifStackTable, and the ipAddrTable.

To compress interfaces that have an administrative status of down, use the **snmp-server interfaces compress-restriction** command.

To compress interfaces according to type, use the **snmp-server interfaces compress** command. To see the list of interfaces that you can remove, use the CLI help:

```
host1(config)#snmp-server interfaces compress ?
  Atm           Atm interface layer
  Atm1483       Atm1483 interface layer
  AtmAal5       AtmAal5 interface layer
  . . .
  SonetVT       SonetVT interface layer
  VlanMajor     VlanMajor interface layer
  VlanSub       VlanSub interface layer <cr>
```

If you enter the **snmp-server interfaces compress** command without keywords, the following interface types are removed from the interface tables:

- ip
- ppp
- ethernetSubinterface
- hdlc
- ipLoopback
- ipVirtual
- pppLinkInterface
- pppoeInterface
- slepInterface/ciscoHdlc

***snmp-server interfaces compress***

- Use to remove interface sublayers from the ifTable, the ifStackTable, and the ipAddrTable.
- Example

  host1(config)#**snmp-server interfaces compress atmAal5**

- Use the **no** version to add interface sublayers to the ifTable, the ifStackTable, and the ipAddrTable.

***snmp-server interfaces compress-restriction***

- Use to exclude interfaces from the ifTable, the ifStackTable, and the ipAddrTable if the administrative status of the interface is down.
- Example

  host1(config)#**snmp-server interfaces compress-restriction ifadminstatusdown**

- Use the **no** version to remove the restriction and allow interfaces with an administrative status of down in the ifTable, the ifStackTable, and the ipAddrTable.

## Controlling Interface Numbering

Each interface in the ifTable is assigned an ifIndex number. RFC 1213 required that ifIndexes use contiguous integers and that the ifIndex be less than the value of the total number of interfaces (ifNumber). More recent RFCs—1573, 2232, and 2863—removed these restrictions to accommodate interface sublayers. The E-series router implementation of SNMP derives index numbers in 32-bit values that are unique on a given router. This numbering scheme can result in large gaps in the ifIndex.

Legacy network management software that was designed to work with RFC 1213 implementations expects contiguous integers and can fail when the software encounters large gaps in the ifIndex.

By default, the router uses a numbering scheme based on RFC 2863. For compatibility with RFC 1213, you can set up the router to use contiguous numbers and to limit the values of the ifIndex and the ifNumber.

***snmp-server interfaces rfc1213***

- Use to set up the interface numbering method in the IfTable to use contiguous integers, which provides compatibility with versions of SNMP that are based on RFC 1213.
- The *maxIfIndex* option sets the maximum value of the ifIndex field that the system will allocate.
- The *maxIfNumber* option sets the maximum number of interfaces allowed in the interface tables.

**CAUTION:** Reducing the value of the maxIfIndex and/or maxIfNumber causes the router to automatically reboot to factory default settings.

- When the IfIndex and IfNumber maximums are reached, the system logs the event and ignores the creation of additional interfaces, which means that new interfaces are not visible in the interface table.
- Example

  host1(config)#**snmp-server interfaces rfc1213 55000 100000**
  WARNING: Execution of this command will cause all configuration settings to revert to factory defaults upon the next system reboot.
  Proceed with 'snmp interfaces rfc1213'? [confirm]

- Use the **no** version to return to the default method of interface numbering.

### Monitoring Interface Tables

Use the following command to view the configuration of your interface tables.

***show snmp group***

- Use to display a list of interface types that are compressed in the interface tables and the interface numbering method configured on the router.
- Field descriptions
  - Compressed(Removed) Interface Types—List of interface types that are removed from the ifTable and ifStackTable
  - Armed Interface Numbering Mode—Interface numbering method configured on the router: RFC1213, RFC2863
  - maxIfIndex—Maximum value that the system will allocate to the ifIndex field
  - maxIfNumber—Maximum number of interfaces allowed in the ifTable
  - Interface Description Setting—Method used to encode the ifDescr and ifName objects: common, legacy, proprietary
- Example

```
host1#show snmp interfaces
Compressed(Removed) Interface Types:
HDLC, FT1, ATM, ATM1483
Armed Interface Numbering Mode:
RFC1213, maxIfIndex=65535, maxIfNumber=65535
Interface Description Setting: proprietary
```

## Configuring Traps

This section provides information for:

- Enabling trap generation
- Setting up filtering of traps by severity
- Configuring trap destinations
- Setting a source address for traps
- Enabling link-status traps
- Specifying an egress point for traps
- Configuring trap queues
- Configuring trap notification logs
- Recovering lost traps

The system generates SNMP traps according to operating specifications defined in supported MIBs.

### IP Hosts

Traps are sent to IP hosts. The IP hosts are configured in a proprietary trap host table maintained by the router (the server). Each entry in the table contains:

- IP address of the trap destination
- Community name (v1 or v2c) or username (v3) to send in the trap message
- SNMP format (v1 or v2) of the notification (trap) PDU to use for that destination
- Types of traps enabled to be sent to that destination
- Trap filters configured for the destination

The maximum number of entries in the SNMP trap host table in each virtual router is eight.

### Trap Categories

The router supports the following trap categories:

- addrPool—Local address pool traps
- atmPing—E-series router proprietary ATM ping traps
- bfdmib—BFD MIB traps
- bgp—BGP state change traps
- bulkstats—Bulk statistics file full and nearly full traps
- cliSecurityAlert—Security alert traps
- dhcp—Dynamic Host Configuration Protocol traps
- dismanEvent—Distributed management (disman) event traps
- dosProtectionPlatform—DoS protection platform traps
- dvmrp—Distance Vector Multicast Routing Protocol traps
- dvmrpProp—E-series router proprietary DVMRP traps
- environment—Power, temperature, fan, and memory utilization traps
- fileXfer—File transfer status change traps
- haRedundancy—High availability and redundancy traps
- inventory—System inventory and status traps
- ip—Internet Protocol traps
- ldp—LDP traps

- link—SNMP linkUp and linkDown traps
- log—System log capacity traps
- mobileIpv4—Mobile IPv4 traps
- mplste—Mplste traps
- mrouter—Mrouter traps
- ntp—E-series router proprietary traps
- ospf—Open Shortest Path First traps
- packetMirror—Packet mirroring traps; packet mirroring–related SNMP categories and traps are visible only to authorized users. See *JUNOSe Policy Management Configuration Guide, Chapter 10, Packet Mirroring Overview* for information about using secure packet mirroring traps.
- pim—Protocol Independent Multicast traps
- ping—Ping operation traps in disman remops (remote operations) MIB
- radius—RADIUS servers fail to respond to accounting and authentication requests traps, or servers return to active service traps
- routeTable—Maximum route limit and warning threshold traps; when this trap is generated, the actual value of the exceeded warning threshold is displayed.
- snmp—SNMP coldStart, warmStart, authenticationFailure; the trap option. The **snmp-server enable traps snmp authentication** command allows customized treatment for SNMP authentication failure traps.
- sonet—SONET traps
- traceroute—Traceroute operation traps (in disman remops MIB)
- trapFilters—Global filters for SNMP trap recipients
- vrrp—Virtual Router Redundancy Protocol traps

To enable global trap categories, use the **snmp-server enable traps** command. To enable trap categories for a specific host, use the **snmp-server host** command.

## Trap Severity Levels

The router provides a method of filtering traps according to severity. Table 23 describes the supported severity levels.

**Table 23: Trap Severity Descriptions**

| Severity Number | Severity Name | System Response |
|---|---|---|
| 0 | Emergency | System unusable |
| 1 | Alert | Immediate action needed |
| 2 | Critical | Critical conditions exist |
| 3 | Error | Error conditions exist |
| 4 | Warning | Warning conditions exist |
| 5 | Notice | Normal but significant conditions exist |
| 6 | Informational | Informational messages |
| 7 | Debug | Debug messages |

You can set up a global filter to filter all traps and/or set up a filter for each host. Trap filters work as follows:

1. An event is posted to the SNMP agent.
2. The system determines whether the corresponding trap category is globally enabled and whether the trap meets the minimum global severity level.
    a. If the trap does not meet these criteria, the system discards the trap.
    b. If the trap does meet these criteria, the trap is handed to the trap host processor.
3. The trap host processor determines whether the trap category is enabled on the host and whether the trap meets the minimum severity level set for the host.
    a. If the trap does not meet these criteria, the system discards the trap.
    b. If the trap does meet these criteria, the trap is sent to the trap recipient.

To set up global severity filters, use the **snmp-server enable traps** command. To set up a severity filter for a specific host, use the **snmp-server host** command.

### *snmp-server enable traps*

- Use to enable and configure SNMP trap generation on a global basis.
- Traps are unsolicited messages sent from an SNMP server (agent) to an SNMP client (manager).
- You can enable the traps listed in *Trap Categories* on page 154.
- You can filter traps according to the trap severity levels described in Table 23 on page 156.

- If you do not specify a trap option, all options are enabled or disabled for the trap type.
- Example

  host1(config)#**snmp-server enable traps atmPing trapfilters critical**

- Use the **no** version to disable SNMP trap generation.

***snmp-server host***

- Use to configure an SNMP trap host to refine the type and severity to traps that the host receives.
- A trap destination is the IP address of a client (network management station) that receives the SNMP traps.
- You can configure up to eight trap hosts on each virtual router.
- You can enable the traps listed in *Trap Categories* on page 154.
- You can filter traps according to the trap severity levels described in Table 23 on page 156.
- Example

  host1(config)# **snmp-server host 126.197.10.5 version 2c westford udp-port 162 snmp link trapfilters alert**

- Use the **no** version to remove the specified host from the list of recipients.

***snmp-server trap-source***

- Use to specify the interface whose IP address is used as the source address for all SNMP traps.

**NOTE:** When there are multiple IP addresses configured on the IP interface that is chosen as the SNMP trap source, the SNMP agent automatically uses the primary IP address of the interface as the SNMP source address on SNMP traps.

- Example

  host1(config)#**snmp-server trap-source fastethernet 0/0**

- Use the **no** version to remove the interface from the trap configuration.

***snmp trap ip link-status***

- Use to enable link-status traps on an IP interface.
- Example

  host1(config-if)#**snmp trap ip link-status**

- Use the **no** version to disable link-status traps on an IP interface.

***snmp trap link-status***

- Use to configure the SNMP link-status traps on a particular interface.
- A *link-up* trap recognizes that a previously inactive link in the network has come up.
- A *link-down* trap recognizes a failure in one of the communication links represented in the server's configuration.
- Example

  host1(config-controll)#**snmp trap link-status**

- Use the **no** version to disable these traps for the interface.

**NOTE:** This command operates in Controller Configuration mode. It is supported only by the DS3, DS1, and FT1 interface layers.

***traps***

- Use to specify traps for OSPF.
- Example

  host1(config-router-rn)#**traps all**

- Use the **no** version to delete the specified trap, group of traps, or all traps.

**NOTE:** For additional information about configuring OSPF-specific traps, see *JUNOSe IP, IPv6, and IGP Configuration Guide, Chapter 5, Configuring OSPF*.

## *Specifying an Egress Point for SNMP Traps*

You can enable SNMP trap proxy, which allows you to specify a single SNMP agent as the egress point for SNMP traps from all other virtual routers. This feature removes the need to configure a network path from each virtual router to a single trap collector.

You can enable SNMP trap proxy from either SNMP or the CLI. Only one SNMP trap proxy can exist for a physical router.

The SNMP trap proxy does not forward global traps that it receives from other virtual routers. The corresponding SNMP agent handles global traps locally and does not forward them to the SNMP trap proxy.

To configure the SNMP trap proxy:

1. Access the virtual router context.
2. Enable or disable the SNMP trap proxy.

***snmp-server trap-proxy***

- Use to enable or disable the SNMP trap proxy.
- Example

  host1(config)#**snmp-server trap-proxy enable**

- Use the **no** version to disable the SNMP trap proxy.

## *Configuring Trap Queues*

You can control the SNMP trap egress rate, specify the method of handling a full queue, and specify the maximum number of traps kept in the queue.

***snmp-server host***

- Use to control the SNMP trap egress rate for the host that is receiving SNMP traps. Use one or more of the following keywords:
  - **drainRate**—Specifies the maximum number of traps per second sent to the host
  - **full**—Specifies the method for handling the queue full condition
  - **size**—Specifies the maximum number of traps kept in the queue
- Example

  host1(config)#**snmp-server host 10.10.10.10 trapqueue drainrate 600 full droplastin size 50**

- Use the **no** version to remove the SNMP host.

## *Configuring Trap Notification Logs*

SNMP uses the User Datagram Protocol (UDP) to send traps. Because UDP does not guarantee delivery or provide flow control, some traps can be lost in transit to a destination address. The Notification Log MIB provides flow control support for UDP datagrams.

You should set up your management applications to periodically request the recorded traps to ensure that the host is up and the management applications have received all the generated traps.

To identify the location of traps logged in the notification log, the system assigns a consecutive index number to each SNMP trap message transmitted from the E-series router. Clients can use the index to detect missing traps.

To configure trap notification logs:

1. Configure the notification log.

   host1(config)**snmp-server notificationlog log 10.10.4.4 adminStatus includeVarbinds**

2. (Optional) Specify when the notification log ages out.

   host1(config)#**snmp-server notificationlog ageout 5**

3. (Optional) Specify the maximum number of entries kept in the notification log.

   host1(config)#**snmp-server notificationlog entrylimit 210**

4. (Optional) Enable the snmpTrap log to severity level info.

   host1(config)#**log severity info snmpTrap**

**NOTE:** Enabling the snmpTrap log provides the same information in the router log as appears in the snmp-server notification log. However, long trap strings may appear truncated.

***log severity***

- Use to set the severity level for a selected category or for systemwide logs.

**NOTE:** For more information about this command, see the *JUNOSe System Event Logging Reference Guide*.

- Example

  host1(config)#**log severity info snmptrap**

- Use the **no** version to return to the default severity value (error) for the selected category. To return all logs to their default severity setting, include an * (asterisk) with the **no** version.

***snmp-server notificationLog ageOut***

- Use to set the ageout for traps in the notification log tables. The range is 0–214748364 minutes.
- Example

  host1(config)#**snmp-server notificationLog ageout 5**

- Use the **no** version to return the ageout limit to the default value, 1440 minutes.

***snmp-server notificationLog entryLimit***

- Use to set the maximum number of notifications kept in all notification log tables.
- The range is 1–500, which means that you can allocate up to 500 notifications across all virtual routers on the router. As you allocate the entry limits for virtual routers, the available range changes to reflect the number of notifications that you have allocated.
- Example

  host1(config)#**snmp-server notificationLog entrylimit 210**

- Use the **no** version to return the limit to the default value, 500.

***snmp-server notificationLog log***

- Use to configure SNMP notification log tables.
- Use the **adminStatus** keyword to enable administrative status.
- Use the **includeVarbinds** keyword to include log names and log indexes in the trap's variable bindings.
- Example

  host1(config)**snmp-server notificationLog log 10.10.4.4 adminStatus includeVarbinds**

- Use the **no** version to remove the notification log configuration.

## *Recovering Lost Traps*

SNMP traps can be lost during startup of the E-series router for one of the following reasons:

1. The SNMP agent begins sending SNMP traps to the host before the line module is initialized.

2. If the SNMP proxy virtual router is initialized after other virtual routers, traps generated by the other virtual routers and sent to the proxy router are lost.

To recover SNMP traps that are lost during system startup, the SNMP agent pings the configured trap host to identify that there is a communication path between E-series router and host. On successful ping acknowledgment, the lost traps are reconstructed for each virtual router. In the case of scenario 1, the reconstructed traps are sent to the proxy virtual router to be routed to the appropriate hosts. In the case of scenario 2, the traps are sent directly to the appropriate hosts.

You can configure the ping timeout window with the **snmp-server host** command. The following are guidelines for setting the maximum ping window:

- If you are losing traps because of scenario 1, base the maximum ping window time on the estimated time that it takes to establish connectivity in a particular network. (For some configurations it can take more than 30 minutes to establish connectivity.)

- If you are losing traps because of scenario 2, we recommend that you use the default value for the maximum ping window time, which is one minute.

***snmp-server host***

- Use to set the ping timeout for the host that is receiving SNMP traps.
- Use the **pingtimeout** keyword to set the ping timeout window; the range is 1–90 minutes.
- Example

  host1(config)#**snmp-server host 10.10.4.4 pingtimeout 2**

- Use the **no** version to remove the SNMP host.

## Configuring the SNMP Server Event Manager

The SNMP server event manager works in conjunction with the Event MIB (RFC 2981). The purpose of this application is to allow many management functions (for example, fault detection, configuration management, accounting management, and performance management). These functions are traditionally performed by the network management station. However, by using the SNMP server event manager, you can distribute some of these functions to E-series routers and automate them.

### *Event MIB Purpose*

The rapid growth of networks has made it impractical to directly manage networks from a single network management station (NMS). This brought about a need for a model that both automated and distributed event management. The goal was to allow devices to monitor themselves and other devices, and to take action under certain conditions.

The Event MIB (RFC 2981) defines a method for creating trigger conditions, testing those conditions, and determining which action to take when a trigger meets those conditions.

The Event MIB allows you to define test conditions for object integers that are accessible in the agent, making it possible to monitor any aspect of a device without defining specific notifications and complicating the agent definition. In this model, because devices have the ability to monitor themselves or other devices, the processing is distributed throughout the network. Also, sending the information only to the NMS that uses an event model reduces both network overhead and processing drain on the NMS.

### *Event MIB Structure*

The Event MIB has three major parts: the trigger table, the objects table, and the event table. These tables also contain subordinate MIB tables that contain more detailed information about the trigger tests.

#### Trigger Table

The trigger table (mteTriggerTable) lists any currently-defined trigger conditions. Triggers fall into three categories—existence, Boolean, and threshold.

An *existence* trigger tests for the existence of a MIB object instance; you can specify that the trigger occur by either the appearance, disappearance, or change in value of a MIB instance.

A *Boolean* trigger tests whether the value of a MIB object (base syntax integer) is equal, unequal, greater than, less than, less than or equal to, or greater than or equal to some defined value.

A *threshold* trigger verifies a MIB object (base syntax integer) in relation to either a rising threshold value, falling threshold value, or both.

You can configure both Boolean and threshold tests to trigger on an *absolute value* or a *delta value* over a determined polling interval.

Subordinate MIB tables exist within the trigger section of each type of trigger test. In other words, each type of trigger (existence, threshold, and Boolean) contains a table that stores added information about that type of trigger test.

For example, a trigger entry of a specific type of test in the mteTriggerTable creates a linked entry in the appropriate subtable. In turn, this subtable contains more specific information about the specific test.

A *delta* table also exists within the trigger tables. This table stores information about any delta values based on any Boolean and threshold triggers. The delta table stores a MIB object that indicates whether any discontinuities occurred for any delta trigger (for example, a router reset).

**NOTE:** When determining discontinuity, the MIB object must be a time-based counter or number. When a polling interval expires and the event agent (router) needs to perform a delta calculation, it first checks the discontinuity MIB object for that trigger. If a discontinuity occurs, the agent does not perform the test for that trigger until the next polling interval.

### Objects Table

The objects table (mteObjectsTable) defines objects that you want to add to event messages. In other words, you can create a list of user-specified objects and bind them to a trigger event. This can provide a snapshot of other values on a router when the trigger occurs. You can bind objects to a specific trigger, a type of test (for example, existence or Boolean tests), or a type of event (for example, rising or falling events).

**NOTE:** This release does not support the objects table.

### Event Table

The event table (mteEventTable) defines what action you want the device to take when a trigger occurs. This action can be in the form of a notification, setting a specified MIB object, or both. The results of these actions are controlled within two subordinate MIB tables—notification and set.

Notifications (mteNotifications), or traps, define what the router sends when an event occurs. These traps include the following:

- When a Boolean or existence trigger occurs, the router sends an mteTriggerFired trap.
- When a rising threshold trigger occurs, the router sends an mteTriggerRising trap.
- When a falling threshold trigger occurs, the router sends an mteTriggerFalling trap.

- If a trigger fails to complete a test for any reason, the router sends a global mteTriggerFailure trap.
- If an event fails to set, the router sends an mteEventSetFailure trap.

Sets define certain modifications to other MIB objects based on a particular event.

## *Configuration Tasks*

To configure the SNMP server event manager:

1. Access the SNMP server management event application.

   host1(config)#**snmp-server management-event**
   host1(config-mgmtevent)#

**NOTE:** You must create a management event instance for each virtual router.

2. (Optional) Specify the maximum number of trigger entries that you want the virtual router to support.

   host1(config-mgmtevent)#**resource 275**

3. Create an event for each trap notification (mteTriggerFailure, mteTriggerFalling, or mteTriggerRising) that you want to use by specifying an event owner and event name.

   host1(config-mgmtevent)#**event sysadmin failuretrigger**
   host1(config-mgmtevent-event)#**exit**
   host1(config-mgmtevent)#**event sysadmin fallingtrigger**
   host1(config-mgmtevent-event)#**exit**
   host1(config-mgmtevent)#**event sysadmin risingtrigger**
   host1(config-mgmtevent-event)#**exit**

**NOTE:** You must create a separate event for each trap notification that you want to use. However, you can specify the trap notification and enable the trap before exiting the event context.

4. Define each event to send a trap notification (mteTriggerFailure, mteTriggerFalling, and mteTriggerRising).

   host1(config-mgmtevent)#**event sysadmin failuretrigger**
   host1(config-mgmtevent-event)#**notification id mteTriggerFailure**
   host1(config-mgmtevent-event)#**exit**
   host1(config-mgmtevent)#**event sysadmin fallingtrigger**
   host1(config-mgmtevent-event)#**notification id mteTriggerFalling**
   host1(config-mgmtevent-event)#**exit**
   host1(config-mgmtevent)#**event sysadmin risingtrigger**
   host1(config-mgmtevent-event)#**notification id mteTriggerRising**
   host1(config-mgmtevent-event)#**exit**

**NOTE:** The mteTriggerFailure notification is a global value. Once you create a failure event notification, it is automatically bound to every trigger with the same owner. If a failure occurs, and the trigger owner and the event owner are the same, the router sends the trap.

5. Enable the event, and exit the event configuration level.

```
host1(config-mgmtevent)#event sysadmin failuretrigger
host1(config-mgmtevent-event)#enable
host1(config-mgmtevent-event)#exit
host1(config-mgmtevent)#event sysadmin fallingtrigger
host1(config-mgmtevent-event)#enable
host1(config-mgmtevent-event)#exit
host1(config-mgmtevent)#event sysadmin risingtrigger
host1(config-mgmtevent-event)#enable
host1(config-mgmtevent-event)#exit
host1(config-mgmtevent)#
```

**NOTE:** Once enabled, you cannot edit an event or trigger configuration. To change an enabled event or trigger, you must delete it and re-create it.

6. Define the trigger that you want to use for an event by specifying a trigger owner and trigger name.

```
host1(config-mgmtevent)#trigger george trigger1
host1(config-mgmtevent-trigger)#
```

7. Specify a MIB object to sample.

```
host1(config-mgmtevent-trigger)#sample value-id 1.3.6.1.2.1.60.1.2.1.1.7
```

8. Specify the frequency (in seconds) at which you want the sampling to occur.

```
host1(config-mgmtevent-trigger)#frequency 100
```

**NOTE:** Unless you specify that you want to perform delta sampling, the values are absolute.

9. (Optional) Specify that you want to perform delta sampling on the sample value ID.

```
host1(config-mgmtevent-trigger)#delta-sampling
```

- (Optional) Enter the discontinuity MIB value ID that you want to test.

  host1(config-mgmtevent-trigger)#**delta-sampling discontinuity-id 1.3.6.1.2.1.31.1.1.1.19.9**

- (Optional) Enter the discontinuity type (timeStamp or timeTicks) that you want the test to use.

  host1(config-mgmtevent-trigger)#**delta-sampling discontinuity-id-type timeStamp**

10. (Optional) Configure the desired SNMP security level for the agent that you want to poll.

    host1(config)#**snmp security read**

11. Define the test values that you want this trigger to use.

    You can define a Boolean test, existence test, or threshold test. See the following sections for procedures.

### Defining a Boolean Test

You can configure a Boolean trigger to test whether the value of an integer object is equal, unequal, greater than, less than, less than or equal to, or greater than or equal to some defined value.

To define a Boolean test:

1. Define the Boolean-test comparison that you want this trigger to use.

   host1(config-mgmtevent-trigger)#**boolean-test comparison greater**

2. (Optional) Specify that you do not want the Boolean test to perform a comparison when this trigger first becomes active.

   host1(config-mgmtevent-trigger)#**boolean-test startup**

3. Specify the events that you want the Boolean-test trigger to use by entering an event owner name and event name.

**NOTE:** You do not need to bind a failure event to a trigger. If you create a failure event and a failure occurs, the router sends the trap if the event owner is the same as the trigger owner.

   host1(config-mgmtevent-trigger)#**boolean-test event george trigger1**

   When specifying an event, use the exact owner name and event name.Specify the Boolean value to which the test compares.

   host1(config-mgmtevent-trigger)#**boolean-test value 5175438**

4. Specify the agent on which the object resides.

   host1(config-mgmtevent-trigger)#**agent context-name router1**

You can obtain the agent context name for a virtual router from the **show snmp agent** command. The agent context name is independent of the virtual router name.Enable the trigger.

host1(config-mgmtevent-trigger)#**enable**

Once enabled, you cannot edit an event or trigger configuration. To change an enabled event or trigger, you must delete it and re-create it.

### Defining an Existence Test

An existence test looks for the existence of a MIB object. The appearance, disappearance, or a change in value of the object can trigger the existence test.

To define an existence test:

1. Define the existence test test-type value that you want this trigger to use.

   host1(config-mgmtevent-trigger)#**existence-test test-type changed**

2. Define the startup threshold condition—absent or present—that you want this trigger to use.

   host1(config-mgmtevent-trigger)#**existence-test startup absent**

3. Specify the events that you want the existence-test trigger to use by entering an event owner name and event name.

**NOTE:** You do not need to bind a failure event to a trigger. If you create a failure event, if a failure occurs, and if the trigger owner and the event owner are the same, the router sends the trap.

   host1(config-mgmtevent-trigger)#**boolean-test event george trigger1**

   When specifying an event, make sure to use the exact owner name and event name.

4. Specify the agent on which the object resides.

   host1(config-mgmtevent-trigger)#**agent context-name router1**

   You can obtain the agent context name for a virtual router from the **show snmp agent** command. The agent context name is independent of the virtual router name.

5. Enable the trigger.

   host1(config-mgmtevent-trigger)#**enable**

   Once enabled, you cannot edit an event or trigger configuration. To change an enabled event or trigger, you must delete it and re-create it.

### Defining a Threshold Test

To define a threshold test:

1. Define the threshold-test values that you want this trigger to use.

**NOTE:** The rising value must always be larger than the falling value. Entering a lower rising value than a falling value will provide invalid results or errors.

   - absolute-value—Use when defining absolute threshold values

     host1(config-mgmtevent-trigger)#**threshold-test absolute-value rising 2000 falling 1900**

   - delta-value—Use when defining delta threshold values

     host1(config-mgmtevent-trigger)#**threshold-test delta-value rising 2000 falling 1900**

2. Define the startup threshold condition that you predict the sample to initially follow—falling, rising, risingorfalling. For example, if you are sampling a MIB value that you know will start from zero and rise, you would specify a rising startup condition.

   host1(config-mgmtevent-trigger)#**threshold-test startup rising**

3. Specify the events (rising or falling) that you want the threshold-test trigger to use by entering an event owner name and event name.

**NOTE:** You do not need to bind a failure event to a trigger. If you create a failure event, if a failure occurs, and if the trigger owner and the event owner are the same, the router sends the trap.

   host1(config-mgmtevent-trigger)#**threshold-test event falling sysadmin fallingtrigger**
   host1(config-mgmtevent-trigger)#**threshold-test event rising sysadmin risingtrigger**

   When specifying an event, make sure to use the exact owner name and event name.

4. Specify the agent on which the object resides.

   host1(config-mgmtevent-trigger)#**agent context-name router1**

   You can obtain the agent context name for a virtual router from the **show snmp agent** command. The agent context name is independent of the virtual router name.

5. Enable the trigger.

   host1(config-mgmtevent-trigger)#**enable**

   Once enabled, you cannot edit an event or trigger configuration. To change an enabled event or trigger, you must delete it and re-create it.

***agent context-name***

- Use to specify the virtual router SNMP agent on which you want to poll MIB objects.
- The default is the current context (virtual router).
- The *contextName* value is the virtual router number in the order the virtual router was created (for example, router1, router2, and so on). Use the **show snmp agent** command to obtain the context name for the virtual router.
- Use the **wildcard** keyword to specify that the context name is a wildcard value.

**NOTE:** Use caution when assigning wildcards. Wildcards can rapidly use up trigger resources.

- Use the **limit** keyword to specify the maximum number of agents to be polled.
- Example 1

  host1(config-mgmtevent-trigger)#**agent context-name router1 wildcard**

- Example 2

  host1(config-mgmtevent-trigger)#**agent context-name router1 wildcard limit 15**

**NOTE:** SNMP server security defaults to “no access.” When using a separate virtual router, you must use the **snmp-server security** command and provide “read” or “read-write” access to other virtual routers.

- Use the **no** version to return to the default context (virtual router).

***boolean-test***

- Use to define Boolean test values for the trigger that you are configuring, including comparison settings, a Boolean value, a startup condition, and binding an event to the Boolean-test trigger.
- Example 1—Specifying a comparison setting

  host1(config-mgmtevent-trigger)#**boolean-test comparison less**

- Example 2—Specifying a Boolean value to which the test compares

  host1(config-mgmtevent-trigger)#**boolean-test value 5175438**

- Example 3—Binding an event to the Boolean-test trigger

  host1(config-mgmtevent-trigger)#**boolean-test event sysadmin booleanTrigger**

- Example 4—Setting the trigger to not perform a Boolean test on startup

  host1(config-mgmtevent-trigger)#**boolean-test startup**

- Use the **no** version to delete the Boolean-test values for this trigger or to remove either the startup condition or event binding.

***delta-sampling***

- Use to specify delta sampling for the trigger you are configuring.
- Example

  host1(config-mgmtevent-trigger)#**delta-sampling**

- (Optional) Use the **discontinuity-id** option to specify a discontinuity MIB ID for the sample. The discontinuity MIB ID monitors the sample for any discontinuity errors during the sample frequency. If a discontinuity error occurs, the router removes the sampling for that interval.
- (Optional) Use the **discontinuity-id-type** option to specify a discontinuity ID type (either timeStamp or timeTicks). The discontinuity ID type indicates the time value that you expect for a specific sample.
- Use the **no** version to turn off delta sampling and use absolute sampling (the default).

***enable***

- Use to enable an event configuration or trigger configuration.
- Example 1—Event Configuration Mode

  host1(config-mgmtevent-event)#**enable**

- Example 2—Trigger Configuration Mode

  host1(config-mgmtevent-trigger)#**enable**

- Once enabled, you cannot edit an event or trigger configuration (even when it is disabled). To change an enabled event or trigger, you must delete it and re-create it.
- There is no **no** version.

***event***

- Use to create an event and access the event configuration mode of the SNMP server event manager.
- Example

  host1(config-mgmtevent)#**event sysadmin failuretrigger**
  host1(config-mgmtevent-event)#

- To leave the event configuration mode, use the **exit** command.
- Use the **no** version to remove the event.

***existence-test***

- Use to define existence test values for the trigger that you are configuring, including binding an event to the existence-test trigger, specifying a startup condition, and defining an existence-test type.
- You can specify one or both startup conditions in the same command. You can specify one, two, or all three test types in the same command.
- Example 1—Binding an event to the Boolean-test trigger

  host1(config-mgmtevent-trigger)#**existence-test event sysadmin existenceTrigger**

- Example 2—Specifying a startup condition

  host1(config-mgmtevent-trigger)#**existence-test startup present**

- Example 3—Specifying an existence test type

  host1(config-mgmtevent-trigger)#**existence-test test-type absent**

- Use the **no** version to delete the existence-test values for this trigger or to remove either the startup condition or event binding.

***frequency***

- Use to set the frequency (in seconds) at which you want MIB sampling to occur.
- Example

  host1(config-mgmtevent)#**frequency 100**

- Use the **no** version to restore the default frequency value (600 seconds).

***notification id***

- Use to specify a trap notification for an event.
- Example

  host1(config-mgmtevent-event)#**notification id mteTriggerFailure**

- Use the **no** version to remove the notification from the event. Removal returns the notification value to its default (0.0)

***resource***

- Use to specify the total number of triggers that the virtual router allows.

**CAUTION:** When assigning wildcards, make sure to allow for enough trigger resources.

- Example

  host1(config-mgmtevent-event)#**resource 250**

- Use the **no** version to restore the default resource value (50).

***sample***

- Use to specify the MIB object that you want to sample for the trigger that you are configuring.
- Example

  host1(config-mgmtevent)#**sample value-id 1.3.6.1.2.1.60.1.2.1.1.7**

- Use the **no** version to remove the MIB object from the trigger. Removal returns the sample value-id to its default (0.0).

***set***

- Use to perform an SNMP set operation under certain event conditions.
- Example—Sets the administrative status of interface 123 to down (2)

  host1(config-mgmtevent-event)#**set context-name router1**
  host1(config-mgmtevent-event)#**set id 1.3.6.1.2.1.2.2.1.7.123**
  host1(config-mgmtevent-event)#**set value 2**

- Use the **no** version to remove the set operation.

***snmp-server management-event***

- Use to launch the SNMP server event manager mode on each virtual router on which you plan to manage events.
- Example

  host1(config)#**snmp-server management-event**
  host1(config-mgmtevent)#

- To leave the SNMP server event manager, use the **exit** command.
- Use the **no** version to delete all the management events.

***snmp-server security***

- Use to specify a security access level for the SNMP agent.
- Example

  host1(config)#**snmp-server security read**

- Use the **no** version to return to the SNMP security level to its default (no-access).

***threshold-test***

- Use to define the threshold values for the trigger that you are configuring, including specifying rising and falling values, a startup threshold condition, and binding an event to the threshold-test trigger.
- Example 1—Specifying absolute values

  host1(config-mgmtevent-trigger)#**threshold-test absolute-value rising 2000 falling 1900**

- Example 2—Specifying a startup threshold condition

  host1(config-mgmtevent-trigger)#**threshold-test startup rising**

- Example 3—Binding an event to the threshold-test trigger

  host1(config-mgmtevent-trigger)#**threshold-test event sysadmin failureTrigger**

- Use the **no** version to delete the threshold-test values for this trigger or remove either the threshold startup condition or event binding.

***trigger***

- Use to create a trigger and access the trigger configuration mode of the SNMP server event manager.
- Example

  host1(config-mgmtevent)#**trigger fred trigger1**
  host1(config-mgmtevent-trigger)#

- To leave the trigger configuration mode, use the **exit** command.
- Use the **no** version to remove the trigger.

## *Monitoring Events*

To view the status of the SNMP agent, use the following **show snmp agent** command. To view statistics associated with events, resources, and triggers, use the **show snmp management-event** command.

***show snmp agent***

- Use to view SNMP MIB agent information.
- Field descriptions
  - context name—Router that contains the MIB agent
  - access permission level—Access permission level for other virtual routers that may want to access the agent
- Example

```
host1#show snmp agent
context name: router1
access permission level: no access
snmp proxy: enabled
```

***show snmp management-event***

- Use to view statistical SNMP event information for event table entries, router resources, and trigger table entries.
- Omit the **events**, **resource**, **statistics**, or **triggers** options to obtain a full output.
- Field descriptions
  - Resource
    - SampleMinimum—Minimum number of samples to be taken
    - SampleInstanceMaximum—Maximum number of samples to be taken
    - SampleInstances—Number of sample instances being monitored

- SampleInstancesHigh—Highest number of samples taken for any of the sample instances
- SampleInstancesLacks—Number of times this system could not take a new sample because that allocation would have exceeded the limit set by mteResourceSampleInstanceMaximum

- Triggers
  - Owner—Owner value assigned to the trigger
  - Name—Name value assigned to the trigger
  - Test—Type of trigger test to perform
  - SampleType—Type of sampling (absolute or delta) to perform
  - ValueID—Object ID of the MIB sample for this trigger
  - ValueIDLimit—Not supported in this release; reads as zero
  - ValueIDWildcard—Not supported in this release; reads as False
  - ContextName—Management context (for example, router1) from which to obtain mteTriggerValueID
  - ContextNameRgulrExprssn—Not supported in this release
  - ContextNameLimit—Not supported in this release; reads as zero
  - ContextNameWildcard—Not supported in this release; reads as False
  - Frequency—Frequency at which this trigger is sampled
  - ObjectsOwner—Not supported in this release
  - Objects—Not supported in this release
  - Enabled—State (False [disabled] or True [enabled]) of the trigger
  - EntryStatus—Active/inactive status of the instance
- Boolean
  - Comparison—Comparison value for this trigger
  - Value—Object ID value to which this trigger compares
  - Startup—Whether or not this trigger performs a Boolean test on startup
  - ObjectsOwner—Owner of this object
  - Objects—Name of this object
  - EventOwner—Owner of this event
  - Event—Name of this event
- Existence
  - Test—Test type for this trigger
  - Startup—Startup condition for this trigger
  - ObjectsOwner—Owner of this object
  - Objects—Name of this object
  - EventOwner—Owner of this event
  - Event—Name of this event

- Statistics
  - trigger owner—Owner value assigned to the trigger
  - trigger name—Name value assigned to the trigger
  - current time—Current UTC time
  - started sampling—UTC time sampling started
  - last sampled—UTC time event last sampled
  - sample instances—Number of sample instances being monitored
  - times sampled—Number of times events sampled
  - event traps—Number of traps sent
  - event sets—Number of events set
  - failures—Number of event failures
  - sample overrun—Number of times the event manager missed sampling for the current value within the given time period
  - failure traps—Number of failure traps sent as a result of event failures
- Threshold
  - Startup—Startup threshold condition for this trigger
  - Rising—Rising threshold condition for this trigger
  - Falling—Falling threshold condition for this trigger
  - DeltaRising—Delta rising threshold condition for this trigger
  - DeltaFalling—Delta falling threshold condition for this trigger
  - ObjectsOwner—Not supported in this release
  - Objects—Not supported in this release
  - RisingEventOwner—Rising event owner value for this trigger
  - RisingEvent—Rising event name value for this trigger
  - FallingEventOwner—Falling event owner value for this trigger
  - FallingEvent—Falling event name value for this trigger
  - DeltaRisingEventOwner—Delta rising event owner value for this trigger
  - DeltaRisingEvent—Delta rising event name value for this trigger
  - DeltaFallingEventOwner—Delta falling event owner value for this trigger
  - DeltaFallingEvent—Delta falling event name value for this trigger
- Delta
  - DiscontinuityID—Discontinuity MIB ID for this trigger
  - DiscontinuityIDWildcard—Not supported in this release
  - DiscontinuityIDType—Discontinuity ID type for this trigger

- Events
  - Owner—Owner value for this event
  - Name—Name of this event
  - Actions—Action (for example, notification) that takes place when this event is triggered
  - Enabled—Enabled state (True [enabled] or False [disabled]) of this event
  - EntryStatus—Entry status for this event
- Notification
  - Notification—Notification trap setting for this event
  - ObjectsOwner—Not supported in this release
  - Objects—Not supported in this release
- Set
  - Object—Object ID that the trigger is setting
  - ObjectWildcard—Whether or not the object is a wildcard
  - Value—Value to which you are setting the object ID when the trigger fires
  - ContextName—Management context (for example, router1) from which to obtain mteTriggerValueID
  - ContextNameWildcard—Whether or not the context name is a wildcard

- Example

```
host1#show snmp management-event
                                  Resource
------------------------------------------------------------------------------
SampleMinimum: 1
SampleInstanceMaximum: 50
SampleInstances: 14
SampleInstancesHigh: 14
SampleInstancesLacks: 0

                                   Triggers
------------------------------------------------------------------------------
Owner: unitTest
Name: booleantest1
Test: boolean
SampleType: absoluteValue
ValueID: 1.3.6.1.2.1.92.1.1.2.0
ValueIDLimit: 0
ValueIDWildcard: False
ContextName: router1
ContextNameLimit: 0
ContextNameWildcard: False
Frequency: 40
ObjectsOwner: unitTest
Objects: test3
Enabled: False
EntryStatus: createAndWait
----------------------- Boolean
Comparison: equal
```

```
Value: 300
Startup: False
ObjectsOwner:
Objects:
EventOwner: unitTest
Event: eventTest1
-------------------------------- Trigger
Owner: unitTest
Name: booleantest2
Test: boolean
SampleType: absoluteValue
ValueID: 1.3.6.1.2.1.92.1.1.2.0
ValueIDLimit: 0
ValueIDWildcard: False
ContextName: router1
ContextNameLimit: 0
ContextNameWildcard: False
Frequency: 40
ObjectsOwner: unitTest
Objects: test3
Enabled: False
EntryStatus: createAndWait
------------------------ Existence
Test: absent
Startup: absent
ObjectsOwner: unitTest
Objects: test3
EventOwner: unitTest
Event: eventTest3
------------------------ Threshold
Startup: falling
Rising: 200
Falling: 100
DeltaRising: 0
DeltaFalling: 0
ObjectsOwner:
Objects:
RisingEventOwner: unitTest
RisingEvent: eventTest2
FallingEventOwner: unitTest
FallingEvent: eventTest3
DeltaRisingEventOwner:
DeltaRisingEvent:
DeltaFallingEventOwner:
DeltaFallingEvent:
------------------------ Delta
DiscontinuityID: 1.3.6.1.2.1.92.1.1.2
DiscontinuityIDWildcard: True
DiscontinuityIDType: timeTicks

                                    Objects
--------------------------------------------------------------------------------
Owner: unitTest Name: test1
Index: 1
ID: 1.3.6.1.2.1.11.1.0
IDWildcard: False
EntryStatus: active
Index: 2
ID: 1.3.6.1.2.1.11.2.0
IDWildcard: False
EntryStatus: active
Index: 5
ID: 1.3.6.1.2.1.11.30.0
```

```
IDWildcard: False
EntryStatus: active

                                    Events
------------------------------------------------------------------------------
Owner: unitTest
Name: eventTest1
Actions: notification set
Enabled: True
EntryStatus: active
------------------------ Notification
Notification: mteTriggerFired
ObjectsOwner: unitTest
Objects: test3
------------------------ Set
Object: 1.3.6.1.2.1.11.1.0
ObjectWildcard: False
Value: -20
ContextName: router
ContextNameWildcard: True
```

## Collecting Bulk Statistics

The router offers an efficient data collection and transfer facility for accounting applications. The E-series router SNMP MIBs extend the accounting data collection mechanism defined in the Accounting-Control-MIB (RFC 2513) to include support for connectionless networks.

Service providers need reasonably accurate data about customers' use of networks. This data is used for billing customers and must be available at a customer's request. Accounting applications based on SNMP polling models consume significant network bandwidth because they poll large volumes of data frequently.

Unfortunately, SNMP is not well suited for gathering large volumes of data, especially over short time intervals. It is inadequate for use by accounting applications because:

- The SNMP PDU layout has a low payload-to-overhead ratio.
- Processing SNMP PDUs is expensive because objects and tables need to be sorted in lexicographic order.

The router avoids the need for continuous polling of SNMP statistics by using applications known as *collectors* to retrieve data. You can configure up to six collectors. The router sends collected statistics through FTP to assigned hosts, known as *receivers*. You must assign a primary receiver to each collector, and you can assign a secondary receiver for redundancy.

**NOTE:** The basic-encoding-rules (BER)–encoding choice is not supported.

You can collect interface bulk statistics based on sets of virtual router groups. If sets of virtual router groups generally correspond to ISPs, you can then forward the relevant data to a particular ISP.

To configure a collector to include data from a specific list of virtual routers, you must first configure a collector and then associate a router set with it. A collector can have up to 64 virtual routers associated with it.

To collect bulk statistics for a subset of all configured subinterfaces, you can define the subinterfaces using the following syntax:

*Slot/Port*[.*subInterfaceId*]

Per virtual router collection is supported on the if-stats and igmp schemas. It is supported on all interface types supported by BulkStats. Collectors modified to use per virtual router collection or configured after a collector has started have a time delay (up to the configured time in seconds) until an active collector starts again.

The maximum number of interfaces for each type of interface and line module can differ. Bulk statistics can collect these statistics when you configure the slots with their respective interfaces to the corresponding maximum values. For information about maximum values see *JUNOSe Release Notes*, *Appendix A*, *System Maximums*.

**NOTE:** Define all interface types before you map a collector to the if-stats schema to ensure that you display statistics for all configured interfaces in the first interval.

The name of the bulk statistics file that is transferred to the host when there is a collectorSequence attribute in the remote name is as follows:

*fileName-z-mmddHHMM-s*.sts

where:

- *fileName*—Name of the file, which includes sysName, sysUpTime, depending on the attributes specified
- *-z*—Receiver index value
- *mmddHHMM*—Timestamp when the receiver is created in month/day/hour/minute format
- *-s*—Actual sequence number

## Interface Strings

Bulk statistics provides interface strings as described in Table 24.

**Table 24: Interface Strings**

| Type of Interface | Common Description Format-Mode Disabled | Common Description Format-Mode Enabled |
|---|---|---|
| IP interfaces | IP | Ip |
| PPP interfaces | PPP | Ppp |
| DS0 interfaces | Ds0 | Ds0 |
| DS1 interfaces | SERIAL | Ds1 |

**Table 24: Interface Strings (continued)**

| Type of Interface | Common Description Format-Mode Disabled | Common Description Format-Mode Enabled |
|---|---|---|
| DS3 interfaces | SERIAL | Ds3 |
| Frame Relay Major interfaces | FR | FrameRelayMajor |
| Ethernet interfaces | ENET | Ethernet |
| Sonet interfaces | SONET | Sonet |
| Sonet Path interfaces | SONET | SonetPath |
| ATM interfaces | ATM | Atm |
| ATM AAL5 interfaces | ATM | AtmAal5 |
| ATM 1483 interfaces | ATM | Atm1483 |
| Ft1 interfaces | SERIAL | Ft1 |
| HDLC interfaces | HDLCIntf | HDLC |
| IpLoopback interfaces | Loopback | IpLoopback |
| IpVirtual interfaces | IpVirtual | IpVirtual |
| Frame Relay Sub interfaces | FR | FrameRelaySub |
| PppOE Major interfaces | PPPoE | PppoeMajor |
| PppOE Sub interfaces | PPPoE | PppoeSub |
| Bridged Ethernet | BRG-ET | BridgedEthernet |
| L2TP Tunnel | L2TP | L2tpTunnel |
| L2TP Session | L2TP | L2tpSession |
| PppLink interfaces | MLPPP | PppLink |
| HDLC interfaces | HDLCEncaps | Hdlc |
| L2TP Destinataion | L2TP | L2tpDestination |
| MPLS Major interfaces | MplsIfMajor | MplsMajor |
| MPLS Minor interfaces | MplsIfMinor | MplsMinor |
| Ppp Network interfaces | MLPPP | PppNetwork |
| Ethernet Sub interfaces | ENET | EthernetSub |
| MultiLink Frame Relay interfaces | MLFR | MultilinkFrameRelay |
| Ip Tunnel Interfaces | IP-TUNNEL | IpTunnel |
| Server Port Interfaces | ServerPort | ServerPort |
| Sonet VT interfaces | SONET | SonetVT |
| Vlan major interfaces | VLAN-MAJ | VlanMajor |
| Vlan sub interfaces | VLAN-SUB | VlanSub |
| Gtp interfaces | Gtp | Gtp |
| L2fTunnel interfaces | L2fTunnel | L2fTunnel |
| L2fSession interfaces | L2fSession | L2fSession |
| L2fDestination interfaces | L2fDestination | L2fDestination |
| IpSec Tunnel interfaces | IpSecTunnel | IpsecTunnel |
| Sg interfaces | SgInterface | SgInterface |
| MPLS L2 Shim interfaces | MplsL2Shim | MplsL2Shim |

**Table 24: Interface Strings (continued)**

| Type of Interface | Common Description Format-Mode Disabled | Common Description Format-Mode Enabled |
|---|---|---|
| MPLS VC Sub interfaces | MplsL3Shim | MplsVcSub |
| LacGen interfaces | LacGen | LacGen |
| Bridge interfaces | BridgeIf | Bridge |
| IpSec Transport interfaces | IPSecTransportIf | IpsecTransport |
| IPv6 interfaces | IPv6If | Ipv6 |
| IPv6 Tunnel interfaces | IPv6TunnelIf | Ipv6Tunnel |
| IPv6 loopback interfaces | IPv6LoopbackIf | Ipv6Loopback |
| OSI interfaces | Osi | Osi |
| LAG interfaces | Lag | Lag |
| Ip Tunnel MDT interfaces | IpTunnelMdt | IpTunnelMdt |

## *Understanding Counter Discontinuity*

Interface counter discontinuity can occur when a counter wraps or after a line module is reloaded or reset. If one of these actions occurs, applications that utilize the counters in expressions or calculations generate erroneous values and misleading graphs.

Because counters are 64 bits long, the possibility of a counter's wrapping naturally would occur so infrequently (for example, in many hundreds of years) that this scenario is not recognized as an issue.

Counter discontinuity does occur, however, when you reload or reset a line module. To indicate reloading or resetting, bulk statistics files contain a record similar to the following:

```
{Controller down slot 3, TUE OCT 29 2004 14:25:10.370 UTC}
```

This record provides a mechanism by which applications can detect discontinuity events. To take advantage of this detection capability, the bulk statistics parsing entity should use the record to terminate expression or formula calculations for the indicated slot and to establish a new baseline.

### *Configuring Collectors and Receivers*

To configure the router to collect statistics:

1. Add names to the FTP host table for the primary and secondary (optional) receivers.

   See *Copying and Redirecting Files* in *Chapter 5, Managing the System*, for information about adding names to the host table.

2. Specify the type of interface on which you want to collect statistics.

   host1(config)#**bulkstats interface-type ppp collector 2**

3. Specify the parameters for the receivers.

   host1(config)#**bulkstats receiver 1 remote-name js:/ftptest/bulk%s%s.sts sysName sysUpTime**

4. Assign the data collector.

   host1(config)#**bulkstats collector 2**

5. Specify the method for data collection.

   host1(config)#**bulkstats collector 2 collect-mode auto**

6. Assign the primary receiver.

   host1(config)#**bulkstats collector 2 primary-receiver 1**

7. (Optional) Assign the secondary receiver.

   host1(config)#**bulkstats collector 2 secondary-receiver 5**

8. (Optional) Specify the time for which the system transfers data.

   host1(config)#**bulkstats collector 2 interval 1000**

9. (Optional) Set the maximum size of the bulk statistics file.

   host1(config)#**bulkstats collector 2 max-size 20480**

10. (Optional) Add descriptive information to the bulk statistics file.

    host1(config)#**bulkstats collector 2 description customer xyz**

11. (Optional) Set the encoding scheme of the ifDescr and ifName objects.

    host1(config)#**bulkstats interfaces description-format common**

12. (Optional) Set the system to retrieve bulk statistics once only.

    host1(config)#**bulkstats collector 2 single-interval**

13. (Optional) Configure bulk statistics traps.

    host1(config)#**bulkstats traps nearly-full**

14. (Optional) Collect bulk statistics per virtual router.

    host1(config)#**bulkstats virtual-router-group collector 2 routerISP3**

**NOTE:** The bulk statistics feature supports generating files on a per interface basis.

***bulkstats collector***

- Use to assign the data collector.
- Example

  host1(config)#**bulkstats collector 2**

- Use the **no** version to delete the collector.

***bulkstats collector collect-mode***

- Use to specify the way the collector retrieves bulk statistics.
- Example

  host1(config)#**bulkstats collector 2 collect-mode auto**

- Use the **no** version to specify that either the user or the system will initiate transfers manually.

***bulkstats collector description***

- Use to add descriptive information to the bulk statistics file.
- Example

  host1(config)#**bulkstats collector 2 description customer xyz**

- Use the **no** version to remove descriptive text from the bulk statistics file.

***bulkstats collector interval***

- Use to specify the time interval in seconds for which the collector transfers data to the receivers.
- Example

  host1(config)#**bulkstats collector 2 interval 1000**

- Use the **no** version to set this time to the default, 360 seconds (6 minutes).

***bulkstats collector max-size***

- Use to set the maximum size of the bulk statistics file for all collectors combined. Even when you configure more than one collector, the first maximum file size configured is the combined size of all collectors.
- The maximum file size that you can configure is 20971520 bytes. However, if you do not configure a maximum size, then the maximum file size defaults to 5767168 bytes.
- Although the CLI accepts the commands, you cannot unconfigure or modify the configuration of the maximum file size until the router is rebooted.
- Example

  host1(config)#**bulkstats collector 2 max-size 20480**

- Use the **no** version to set the size of the bulk statistics file to the default, 5767168 bytes.

***bulkstats collector primary-receiver***

- Use to assign the primary receiver to which the system transfers data.
- The index for the receiver must match the index that you specified with the **bulkstats receiver remote-name** command.
- Example

  host1(config)#**bulkstats collector 2 primary-receiver 7**

- Use the **no** version to clear the primary receiver and disable the collector.

***bulkstats collector secondary-receiver***

- Use to assign the secondary (that is, the backup) receiver to which the system transfers data.
- The index for the receiver must match the index you specified with the **bulkstats receiver remote-name** command.
- Example

  host1(config)#**bulkstats collector 2 secondary-receiver 5**

- Use the **no** version to clear the secondary receiver.

***bulkstats collector single-interval***

- Use to set the system to retrieve bulk statistics once only, rather than periodically.
- Example

  host1(config)#**bulkstats collector 2 single-interval**

- Use the **no** version to set the system to retrieve bulk statistics periodically, the default situation.

***bulkstats interfaces description-format common***

- Use to set the encoding scheme of the ifDescr object that the bulk statistics application reports to the conventional industry method.
- This command provides compatibility with software that uses the industry encoding scheme.
- For more information, see *Configuring Encoding Method* on page 150.
- Example

  host1(config)#**bulkstats interfaces description-format common**

- Use the **no** version to return to the proprietary method of encoding.

***bulkstats interface-type***

- Use to configure the interface or subinterface type on which you want to collect statistics.
- You can provide an interface specifier (location) to identify a specific interface on which you want to collect statistics.
- If you define more than one collector, you must specify a unique collector index, in the range 1–65535.
- The supported interface types are:
  - **atm**—Collects statistics on ATM interfaces
  - **atm1483**—Collects statistics on ATM 1483 interfaces
  - **ethernet**—Collects statistics on Ethernet interfaces
  - **frame-relay**—Collects statistics on Frame Relay interfaces
  - **frame-relay-sub**—Collects statistics on Frame Relay subinterfaces
  - **hdlc**—Collects statistics on Cisco HDLC interfaces
  - **ip**—Collects statistics on IP interfaces
  - **mplsMajor**—Collects statistics on MPLS major interfaces
  - **mplsMinor**—Collects statistics on MPLS minor interfaces
  - **mplsL2shim**—Collects statistics on MPLS shim interfaces
  - **ppp**—Collects statistics on PPP
  - **vlan**—Collects statistics on VLAN subinterfaces

**NOTE:** You cannot collect statistics on the SRP Ethernet interface.

- Example 1

  host1(config)#**bulkstats interface-type ppp 3/1 collector 2**

- Example 2

  host1(config)#**bulkstats interface-type vlan 2/3:1 collector 1**

- Example 3

  host1(config)#**bulkstats interface-type mplsMajor 2/3:1 collector 1**

- Use the **no** version to delete the interface type from bulk statistics collection. Deletion of a particular interface type takes effect at the next collection interval.

***bulkstats receiver remote-name***

- Use to configure the parameters for receivers.
- Bulk statistics transfers require the configuration of a remote FTP server.
- The receivers must appear in the FTP host table. The name of the host must match the name you specify with this command. The hostname is relative to the virtual router's context when you issue this command.
- When specifying the remote filename for bulk statistics, you must precede the filename with the hostname followed by the :/ characters.
- Example

  host1(config)#**bulkstats receiver 1 remote-name js:/ftptest/bulk%s%s.sts sysName sysUpTime**

**NOTE:** The % variables in the remote name are replaced at runtime with the sysName and sysUpTime parameters to produce variable filenames on the remote host.

- Use the **no** version to delete the receiver.

***bulkstats traps***

- Use to configure bulk statistics traps.
- You must configure SNMP correctly and specify a valid trap source. Otherwise, the system will not send SNMP traps.
- Example

  host1(config)#**bulkstats traps nearly-full**

- Use the **no** version to disable the trap.

***bulkstats virtual-router-group***

- Use to collect interface statistics for each virtual router.
- A collector can have a maximum of 64 virtual routers associated with it.
- Routers are identified by their assigned name or router index.
- Supported only on if-stats and igmp schemas.
- Supported on all interface types supported by the bulk statistics application.
- Collectors modified to use per virtual router collection or configured after a collector has started have a time delay until an active collector starts again.

- Example

  host1(config)#**bulkstats virtual-router-group collector *2* routerISP3**

- Use the **no** version to prevent bulkstats from being reported for virtual router groups.

## *Deleting All Bulkstats Configurations*

Although individual bulkstats commands allow you to disable or delete a specific bulkstats parameter, the CLI also allows you to remove all bulkstats configurations from the router at one time.

### *no bulkstats*

- Use to remove all bulkstats configurations from the router at one time.
- Example

  host1(config)#**no bulkstats**

## *Monitoring Collection Statistics*

To view the parameters the router uses to collect statistics, use the following **show bulkstats** commands.

To include or exclude lines of output based on a text string that you specify, use the output filtering feature for **show** commands. For details, see *Chapter 2, Command-Line Interface*.

### *show bulkstats*

- Use to display the bulk statistics data collection configuration.
- Field descriptions
  - AdminStatus—Administrative status of the bulk statistics application
  - OperStatus—Operational status of the bulk statistics application, enabled or disabled
  - Interface Description Setting—Method used to encode the ifDescr object: common, proprietary, industry-common
  - File Format—End of the line format in bulkstats files, carriage return and line feed (CR + LF) or LF
  - Current Time—Current system time used to compare with the collection stop/start time
  - Intervals—Number of times the bulk statistics collector has cycled through a collection
  - PrimaryXfers—Number of times the bulk statistics collector has attempted a data file transfer to a primary server
  - PrimaryFails—Number of primary server transfer failures
  - SecondaryXfers—Number of times the bulk statistics collector has attempted a data file transfer to a secondary server
  - SecondaryFails—Number of secondary server transfer failures

- BulkStats Collector Information:
  - Index—Bulk statistics collector index number
  - CurrSize—Current size of the bulk statistics file in bytes
  - MaxSize—Maximum size configured for the bulk statistics file in bytes
  - Intrvl—Time interval between bulk collections in seconds
  - Mode—How often the collector is set up to collect statistics:
    - periodic—Collects statistics periodically
    - single-interval—Collects statistics once only
  - XferMode—Collect mode configured for the collector:
    - auto—Agent transfers file when interval expires
    - manual—Network management system or the user initiates transfers
    - onFull—Agent transfers file when it reaches the maximum size
  - State
    - inProg—Collector is properly configured and currently active
    - notInSvc—Collector has been decommissioned by a management client
    - notReady—Collector does not have enough configuration information to go active
    - error—Configuration or operational error
  - Index—Bulk statistics collector index number
  - Primary-Receiver—Index number of the primary receiver to which the system transfers data, if defined
  - Second-Receiver—Index of the secondary receiver to which the system transfers data
  - Last Transfer Failure—Last time that the collector attempted to retrieve statistics and was unsuccessful
  - Interval Start Time—Start of current interval of bulk collections. The collector began collecting bulk statistics at this time.
  - Interval Stop Time—End of current interval of bulk collections.
- Schema Information:
  - Index—Index number of the schema
  - Subtree—Type of bulk statistics schema configured on the collector: if-stack, if-stats, or system
  - CollectorIndex—Bulk statistics collector index number
  - Create-Delete Time Stats—State of final statistics collection (enabled or disabled)

- Create-Delete Interface Type—Interface type associated with final statistics collection (ATM 1483, IP, PPP)
- State
  - active—Schema is properly configured and currently active
  - notInSvc—Schema has been decommissioned by a management client
  - notReady—Schema does not have enough configuration information to go active
  - error—Configuration or operational error
- Subtree List—Types of statistics the schema is configured to receive

- Interface Types:
  - Index—Index number of the interface type entry
  - Type—Interface type for which bulk statistics collection is configured
  - CollectorIndex—Index number of the collector to which the interface type applies
  - State
    - active—Interface type is properly configured and currently active
    - notInSvc—Interface type has been decommissioned by a management client
    - notReady—Interface type does not have enough configuration information to go active
    - error—Configuration or operational error

- Receiver Information:
  - Index—Index number of the receiver
  - RemoteFileName—Hostname, path, and filename of the remote FTP server
  - State
    - active—Receiver is properly configured and currently active
    - notInSvc—Receiver has been decommissioned by a management client
    - notReady—Receiver does not have enough configuration information to go active
    - error—Configuration or operational error
  - Status
    - Success
    - Copy source does not exist or is unreachable

        - Copy failed
        - File in use
    - Virtual Router Groups:
        - Collector—Number that identifies the particular data collector, in the range 1–65535
        - Virtual-Routers—Set of virtual router names (up to 64 names)
- Example

```
host1#show bulkstats

  AdminStatus:  enabled
  OperStatus:   enabled
  Interface Description Setting: industry-common
  File Format: CR+LF
Current Time: TUE AUG 15 2002 15:54:20 UTC

  Intervals PrimaryXfers PrimaryFails SecondaryXfers SecondaryFails
  --------- ------------ ------------ -------------- --------------
  0         0            0            0              0

  BulkStats Collector Information:
  Index  CurrSize  MaxSize   Intrvl  Mode      XferMode  State
  -----  --------  --------  ------  --------- --------  -------
  1      490       3670016   600     periodic  manual    inProg
  2      0         3670016   360     periodic  manual    notReady

  Index  Primary-Receiver  Second-Receiver  Last Transfer Failure
  -----  ----------------  ---------------  --------------------
  1      1                 not defined
  2      not defined       not defined

  Index Interval Start Time          Interval Stop Time
  ----- ---------------------------- -------------------------
  1     TUE AUG 15 2000 15:52:33 UTC TUE AUG 15 2000 16:02:33 UTC
  2     Not started                  N/A

Schema Information:
  Index  Subtree
  -----  -------------------------------------------------
  1      ifStats

  Index  CollectorIndex  State
  -----  --------------  --------
  1      1               active

  Index  Create-Delete Time Stats  Create-Delete Interface Types
  -----  ------------------------  -----------------------------
  1      enabled                   IP

  Index  Subtree List
  -----  ------------------------------------------------------------
  1      all
```

```
  Interface Types:
  Index         Type             CollectorIndex  State
  -----  ----------------------  --------------  --------
  1      Ppp                     1               active
  6      Ethernet                1               active
  11     Atm1483                 1               active

  Receiver Information:
  Index  RemoteFileName
  -----  ------------------------------------------------------
  1      host:/upload/bulkStas.sts

  Index  State     Status
  -----  --------  ---------------------------------------------
  1      notReady  Copy source does not exist or is unreachable

Collector  Virtual-Routers
---------  ---------------
33             serviceProviderABC
655             default
```

***show bulkstats collector description***

- Use to display information about the collector's file description.
- Field descriptions
  - Index—Index number of the bulk statistics collector
  - FileDescription—Descriptive information added to the bulk statistics file with the **bulkstats collector description** command
- Example

```
host1#show bulkstats collector description
  Index  FileDescription
  -----  ----------------------
  1      Bulk SNMP Statistics Collection
```

***show bulkstats collector interval***

- Use to display information about the collector transfer interval configuration.
- Field descriptions
  - Index—Index number of the bulk statistics collector
  - Interval—Amount of time, in seconds, that the collector transfers data to the receiver
- Example

```
host1#show bulkstats collector interval
Index  Interval
-----  --------
1      360
```

***show bulkstats collector max-size***

- Use to display information about the bulk statistics maximum file size configuration.
- Field descriptions
  - Index—Index number of the bulk statistics collector
  - MaxSize—Maximum size of the bulk statistics file in bytes
- Example

```
host1#show bulkstats collector max-size
Index  MaxSize
-----  ------------
1      2097152
```

***show bulkstats collector transfer-mode***

- Use to display information about the bulk statistics transfer mode configuration.
- Field descriptions
  - Index—Index number of the bulk statistics collector
  - Transfer-Mode:
    - auto-xfer—Server automatically transfers the bulk statistics files to a remote FTP server
    - manual-xfer—Server expects the user to transfer bulk statistics files
    - on-file-full—Server transfers the bulk statistics file when the file reaches its maximum size
  - Primary-Receiver—Receives the bulk statistics sent by the collector
  - Secondary-Receiver—Serves as a backup to the primary receiver
- Example

```
host1#show bulkstats collector transfer-mode
Index  Transfer-Mode  Primary-Receiver  Secondary-Receiver
-----  -------------  ----------------  ------------------
1      auto-xfer      1                 2
```

***show bulkstats interface-type***

- Use to display information about the bulk statistics interface types configuration.
- Field descriptions
  - Interface Types:
  - Index—Index number of the interface type entry
  - Type—Interface type for which bulk statistics collection is configured

- CollectorIndex—Index of the collector to which the interface type applies
- State
  - active—Interface type is properly configured and currently active
  - notInSvc—Interface type has been decommissioned by a management client
  - notReady—Interface type does not have enough configuration information to go active
  - error—Configuration/operational error

- Example

```
host1#show bulkstats interface-type

Interface Types:
  Index         Type             Collector  State
  -----  ----------------------  ---------  --------
  1      ppp                     1          active
```

### *show bulkstats receiver*

- Use to display information about the remote file configuration of the bulk statistics receiver.
- Field descriptions
  - Index—Index number of the receiver
  - RemoteFileName—Hostname, path, and filename of the remote FTP server
  - Index—Index number of the receiver
  - State
    - active—Receiver is properly configured and currently active
    - notInSvc—Receiver has been decommissioned by a management client
    - notReady—Receiver does not have enough configuration information to go active
    - error—Configuration/operational error
  - Status
    - Success
    - Copy source does not exist or is unreachable
    - Copy failed
    - File in use

- Example

```
host1#show bulkstats receiver

  Index  RemoteFileName
  -----  ---------------------------------------------
  1      f:/upload/bulkStas.sts

  Index  State     Status
  -----  --------  ---------------------------------------------
  1      notReady  Copy source does not exist or is unreachable
```

***show bulkstats statistics***

- Use to display bulk statistics counters.
- Field descriptions
  - AdminStatus—Administrative status of the bulk statistics application
  - OperStatus—Operational status of the bulk statistics application
  - HdwDetects—Number of times the bulk statistics application detected a line module bulkstat collector's presence
  - HdwCollectorCreates—Number of line module collectors created
  - CollectorCreateReqs—Number of times the bulk statistics application requested the creation of a line module collector
  - CollectorStopReqs—Number of times the bulk statistics application requested the line module collectors to stop
  - CollectorDeleteReqs—Number of times the bulk statistics application requested the deletion of a line module collector
  - CollectorStarts—Number of times the bulk statistics collector has started
  - CollectorIncompleteCfgs—Number of times the bulk statistics collector attempted to start a collector, but failed because the collector's configuration was incomplete
  - CollectorStopFailures—Number of times the bulk statistics collector failed during a collector stop request
  - DriverErrors—Number of bulk statistics driver errors
  - FileSizeFulls—Number of times the bulk statistics application ran out of storage space
  - CollectorFileNearlyFullTraps—Number of nearly full events posted to the SNMP agent on this router
  - CollectorFileFullTraps—Number of file full events posted to the SNMP agent on this router
  - Intervals—Number of times the bulk statistics collector has cycled through a collection
  - PrimaryXfers—Number of times the bulk statistics collector has attempted a data file transfer to a primary server
  - PrimaryFails—Number of primary server transfer failures
  - SecondaryXfers—Number of times the bulk statistics collector has attempted a data file transfer to a secondary server

- SecondaryFails—Number of secondary server transfer failures
- BulkStats Collector Statistics:
  - Index—Bulk statistics collector index
  - CurrSize—Current size of the bulk statistics storage file in bytes
  - CreateErrs—Number of bulk statistics collector create errors
  - Last Transfer Failure—Last time that the collector attempted to retrieve statistics and was unsuccessful
  - Interval Start Time—Start of current interval or bulk collections. The collector began collecting bulk statistics at this time.
  - Interval Stop Time—End of current interval of bulk collections
- Dynamic Interface Collector statistics:
  - Collector Index—Bulk statistics collector index
  - Slot#—Slot number from which the statistics were obtained
  - Received—Number of records for dynamic interfaces that were reported by the specified interface
  - Transferred—Number of record for dynamic interface that were written to the bulk statistics (.sts) file.
  - Dropped—Number of records for dynamic interfaces that were dropped (that is, not written to the bulk statistics [.sts] file)

- Example

```
host1#show bulkstats statistics

AdminStatus:                  enabled
OperStatus:                   enabled
HdwDetects:                   4
HdwCollectorCreates:          8
CollectorCreateReqs:          2
CollectorStopReqs:            0
CollectorDeleteReqs:          0
CollectorStarts:              25
CollectorIncompleteCfgs:      3
CollectorStopFailures:        0
DriverErrors:                 0
FileSizeFulls:                0
CollectorFileNearlyFullTraps: 0
CollectorFileFullTraps:       0

  Intervals PrimaryXfers PrimaryFails SecondaryXfers  SecondaryFails
  --------- ------------ ------------ --------------  --------------
  24         18            5             0               0

  BulkStats Collector Statistics:
  Index  CurrSize  CreateErrs  Last Transfer Failure
  -----  --------  ----------  --------------------------
  1      331       0           MON JAN 24 2001 17:21:33 UTC
  2      0         0

  Index Interval Start Time          Interval Stop Time
  ----- ---------------------------- -------------------------
  1     MON JAN 24 2001 19:09:33 UTC MON JAN 24 2001 19:15:33 UTC
  2     Not started                  N/A
```

```
Dynamic Interface Collector statistics:
  CollectorIndex  Slot#  Received    Transferred  Dropped
  --------------  -----  ----------  -----------  ----------
  1               1      0           0            0
```

### *show bulkstats traps*

- Use to display information about the bulk statistics traps configured to collect statistics.
- Field descriptions
  - Trap Type
    - nearly-full—Trap will be posted to the SNMP entity on this system when the threshold is reached
    - file-full—Trap will be posted to the SNMP entity on this system when the trap reaches 100 %
  - State—Configuration setting: enabled, disabled
  - Threshold—Nearly full trap will be posted to the SNMP entity on this system when this percentage is reached
  - Traps Sent—Number of times this event was posted to the SNMP entity on this system
- Example

```
host1#show bulkstats traps
Trap Type    State   Threshold    Traps Sent
-----------  -------  ----------  ----------
file-full    enabled  N/A         0
nearly-full  enabled  5           0
```

### *show bulkstats virtual-routers*

- Use to display information about the bulk virtual router group configuration.
- Field descriptions
  - Collector—Number that identifies the particular data collector, in the range 1–65535
  - Virtual-Routers—Set of virtual router names (up to 64 names)
- Example

```
host1#show bulkstats virtual-routers

Collector  Virtual-Routers
---------  ---------------
33             serviceProviderABC
655             default
```

## *Configuring Schemas*

You can also set a management schema for bulk statistics. A schema is a group of attributes or counters that provide an efficient way to retrieve specific types of information about the router. The bulk statistics application supports five schema configurations: *igmp, if-stack, if-stats*, *policy,* and *system*.

**NOTE:** There are no explicit schema objects for the if-stack and system schemas.

Table 25 shows the type of data each schema retrieves.

**Table 25: Data Retrieved According to Schema**

| Schema | Retrieves |
|---|---|
| igmp | Statistics associated with various IGMP components. |
| if-stack | The interface and interface column configuration. It is a complete retrieval of the ifStackTable, and using it can dramatically reduce the time to discover the configured interfaces and their stacking relationship on a router. |
| if-stats | Usage data on sets of interface types. The interface usage data is the ifTable/ifXTable counters. Note that the ifXTable supports 64-bit counters and the data written into the bulk statistics file supports the 64-bit counters. |
| policy | Statistics associated with a specified policy, a policy type, or traffic tagged by a policy with a color tag. |
| system | Global system and per-module statistics and information. The global system statistics retrieved are the sysUpTime and nvsUtilPct. The per-module statistics and information retrieved include the intPhysicalDesc, the cpuUtilPct, and the memUtilPct. |

### igmp Objects

Table 26 presents igmp objects you can configure using the **bulkstats schema subtree** command.

**Table 26: Schema igmp Objects**

| Object | Definition |
|---|---|
| all | Configure IGMP schema for all attributes |
| dest-address | Configure IGMP schema for destination address |
| igmp-cmd | Configure IGMP schema for IGMP command |
| lower-interface | Configure IGMP schema for lower interface |
| multicast-group | Configure IGMP schema for multicast group |
| router-index | Configure IGMP schema for router index |
| source-address | Configure IGMP schema for source address |
| time-stamp | Configure IGMP schema for time stamp |

### if-stats Objects

Table 27 presents if-stats objects you can configure using the **bulkstats schema subtree** command.

**Table 27: Schema ifStats Objects**

| Object | Definition |
|---|---|
| all | Configure IfStats schema for all stats |
| correlator | Configure IfStats schema for correlator |
| in-bcast-pkts | Configure IfStats schema for in-bcast-pkts |
| in-discards | Configure IfStats schema for in-discards |
| in-errors | Configure IfStats schema for in-errors |
| in-mcast-octets | Configure IfStats schema for in-mcast-octets |
| in-mcast-pkts | Configure IfStats schema for in-mcast-pkts |
| in-octets | Configure IfStats schema for in-octets |
| in-policed-octets | Configure IfStats schema for in-policed-octets |
| in-policed-pkts | Configure IfStats schema for in-policed-pkts |
| in-spoofed-pkts | Configure IfStats schema for in-spoofed-pkts |
| in-ucast-pkts | Configure IfStats schema for in-ucast-pkts |
| in-unknown-protos | Configure IfStats schema for in-unknown-protos |
| lower-interface | Configure IfStats schema for lower-interface |
| out-bcast-pkts | Configure IfStats schema for out-bcast-pkts |
| out-discards | Configure IfStats schema for out-discards |
| out-errors | Configure IfStats schema for out-errors |
| out-mcast-octets | Configure IfStats schema for out-mcast-octets |
| out-mcast-pkts | Configure IfStats schema for out-mcast-pkts |
| out-octets | Configure IfStats schema for out-octets |
| out-policed-octets | Configure IfStats schema for out-policed-octets |
| out-policed-pkts | Configure IfStats schema for out-policed-pkts |
| out-sched-octets | Configure IfStats schema for out-sched-octets |
| out-sched-pkts | Configure IfStats schema for out-sched-pkts |
| out-ucast-pkts | Configure IfStats schema for out-ucast-pkts |
| time-offset | Configure IfStats schema for time-offset |

All the schema if-stats objects in Table 27 apply to both layer 2 and layer 3 interfaces, except usdAcctngSpoofedPkts, which is specific to layer 3.

Defining all interface types before you map a collector to the if-stats schema ensures that you display statistics for all configured interfaces in the first interval.

You can get more accurate rate statistics by using the **time-offset** parameter. To use this parameter you must navigate to the **if-stats subtreelist**. The **time-offset** parameter is included in each bulk statistics interface record and is the offset from the master interval at which the record was collected.

### policy Objects

Table 28 presents policy objects you can configure using the **bulkstats schema subtree** command.

**Table 28: Schema Policy Objects**

| Object | Definition |
|---|---|
| all | Configure policy schema for all statistics |
| green-bytes | Configure policy schema for green bytes |
| green-packets | Configure policy schema for green packets |
| red-bytes | Configure policy schema for red bytes |
| red-packets | Configure policy schema for red packets |
| upper-green-bytes | Configure policy schema for upper green bytes |
| upper-green-packets | Configure policy schema for upper green packets |
| upper-red-bytes | Configure policy schema for upper red bytes |
| upper-red-packets | Configure policy schema for upper red packets |
| upper-yellow-bytes | Configure policy schema for upper yellow bytes |
| upper-yellow-packets | Configure policy schema for upper yellow packets |
| yellow-bytes | Configure policy schema for yellow bytes |
| yellow-packets | Configure policy schema for yellow packets |

***bulkstats schema***

- Use to create the schema for collecting bulk statistics.
- Example

  host1(config)#**bulkstats schema 4**

- Use the **no** version to delete the specified schema.

**NOTE:** If you create a collector but there is no schema for that collector, the collector will not be active, and a schema will be created automatically for that collector to collect if-stats for all subtree attributes.

***bulkstats schema subtree***

- Use to set the schema for collecting data. Specify one of the following keywords:
  - **if-stack**—Retrieves the interface and interface column configuration.
  - **if-stats**—Retrieves interface usage data on sets of interface types; using the **subtreelist** keyword along with the **if-stats** keyword lets you specify specific counters and lets you set the **time-offset** parameter; using the **if-create-delete-time-stats** keyword along with the **if-stats** keyword retrieves interface final statistics (interface statistics that may be lost during higher create or delete frequency) on a per-interface basis.
  - **igmp**—Retrieves IGMP usage data; using the **subtreelist** keyword along with the **igmplist** keyword lets you obtain statistics for one or more specific IGMP lists.
  - **policy**—Retrieves policy usage data.
  - **system**—Retrieves global system and per-module statistics and information.
- Example 1

  host1(config)#**bulkstats schema 1 subtree if-stats subtreelist lower-interface**

- Example 2

  host1(config)#**bulkstats schema 5 subtree if-stats if-create-delete-time-stats interfaceType ?**

```
atm1483     Configure bulkstats for ATM 1483 sub-interfaces
ip          Configure bulkstats for IP interfaces
mplsL2Shim  Configure bulkstats for MPLS shim Interfaces
mplsMajor   Configure bulkstats for MPLS major Interfaces
mplsMinor   Configure bulkstats for MPLS minor Interfaces
ppp         Configure bulkstats for PPP interfaces
vlan        Configure bulkstats for VLAN Sub-Interfaces
```

- Use to collect statistics on a specified policy, a policy type, or based on color-coded tags applied by a policy. Specify one of the following keywords:
  - **policy-name—**Collects statistics for a specified policy
  - **policy-type**—Collects data on input policies, local input policies, output policies, or secondary output policies
  - **policy-subtreelist**—Collects statistics based on color-coded tags applied by a policy
- You create policies using the **policy-list** command. See *JUNOSe Policy Management Configuration Guide, Chapter 1, Managing Policies on the E-series Router*.
- Example

  host1(config)#**bulkstats schema 4 subtree policy policy-name XMYpolicy**

- Use the **no** version to delete the specified schema.

## *Monitoring Schema Statistics*

You are able to display your configuration and monitor the data generated by schemas.

### *show bulkstats schema*

- Use to display data on the bulk statistics schema.
- Field descriptions
  - Schema Information:
  - Index—Index number of the schema
  - Subtree—Type of bulk statistics schema configured on the collector: igmp, if-stack, if-stats, policy, or system
  - CollectorIndex—Bulk statistics collector index (same as the SNMP table index)
  - State
    - active—Schema is properly configured and currently active
    - notInService—Schema has been decommissioned by a management client
    - notReady—Schema does not have enough configuration information to go active
    - error—Configuration/operational error
  - Subtree List—Type(s) of statistics the schema is configured to receive
- Example 1

```
host1#show bulkstats schema

  Schema Information:
  Index  Subtree            CollectorIndex  State
  -----  -----------------  --------------  --------
  1      ifStack            1               active
  2      system             2               active

  Index  Subtree List
  -----  -------------------------------------------------
  1      N/A
  2      N/A
```

- Example 2

```
host1#show bulkstats schema

  Schema Information:
  Index  Subtree            CollectorIndex  State
  -----  -----------------  --------------  --------
  1      ifStats            1               active
  2      system             2               active

  Index  Subtree List
  -----  -------------------------------------------------
  1      ifOutErrors; ifLowerInterface; ifTimeOffset
  2      N/A
```

## Configuring Interface Numbering Mode

E-series routers support the RFC 1213 interface numbering mode on bulkstats. This mode is contrasted with the default interface numbering mode.

The RFC 1213 numbering mode is based on a 32-bit contiguous integer value starting from 1 and ranging to ifNumber. This mode differs from the default interface numbering mode, which encodes a type field in the upper 8 bits of a 32-bit integer. The use of the upper 8 bits creates large gaps in the ifIndex numbering scheme.

There is no re-use of ifIndex values in RFC 1213 mode, whereas in the default interface numbering mode, ifIndex values can be re-used. In the default interface numbering mode, re-use of ifIndex values across reboots is permitted and is basically known as ifIndex re-numbering.

In RFC 1213 mode, however, the interface numbers are not re-used during a single initialization of the device and renumbering of ifIndexes occurs after a system reboot. In the default interface numbering mode, ifIndexes are persistent across system reboots and can be reused without resetting the value of sysUpTime.

In RFC 1213 mode, two parameters control the size of the ifIndex range and the total number of interfaces in the standard interface tables—maxIfIndex and maxIfNumber. There is no such control in the default interface numbering mode.

In RFC 1213 mode, interface creations should not result in gaps in the ifIndex range. A gap that results from the deletion of an interface is acceptable because it is handled by older network management applications. The gaps are eliminated after the router is rebooted. However, in the default interface numbering mode, large gaps occur from the creation of interfaces due to the use of the upper 8 bits of the ifIndex for interface type encoding. Gaps are not eliminated after a system reboot.

In RFC 1213 mode, small gaps can occur in the creation of IP interfaces when virtual routers are used. These gaps are minimized but not eliminated when the router is rebooted.

Rather than seeing an ifIndex value of 1 and 10066329, for example, a management client would see ifIndex values of 1 and 2.

### *bulkstats interfaces rfc1213*

- Use to enable the RFC 1213 interface numbering mode on bulkstats.
- Example

  host1(config)#**bulkstats interfaces rfc 1213**

- Use the **no** version to disable the RFC 1213 interface numbering mode on bulkstats.

## Using the Bulk Statistics Formatter

The bulk statistics formatter allows you to set a remote filename dynamically and specify the format for the end of each line in the bulkstats file.

### *Setting Remote Filenames*

The router supports the following special characters for remote filenames:

- %x—An integer in hexadecimal format (base 16)
- %s—A character string
- %u—An unsigned integer in decimal (base 10)
- %d—An integer in decimal (base 10)

The % variables in the remote name are replaced at runtime with the sysName and sysUpTime parameters to produce variable filenames on the remote host.

See the **bulkstats receiver remote-name** command.

```
host1(config)#bulkstats receiver 1 remote-name "bulk%s%d.sts" sysName
collectorSequence
```

#### Guidelines

The current capabilities and limitations of the bulk statistics formatter are:

- If you add %d or any numeric formatter for a string value (such as sysName), the attribute name will be used (for instance, sysName). The opposite is also true, except for sysUptime, which will use %s as a %u.
- You can use %% if you want a % character to be part of the parsed name.
- You can use the same attribute multiple times. For example, you may want a name that has %x and %u of collectorSequence.
- Currently, there is no control over sequence numbers, except for the guarantee that the formatter will:
  1. Use sequential values, beginning from 1
  2. Persist through system reboot
- If you need the sequential number to restart, remove and then add the bulk statistics receiver again.
- You can use up to 128 characters for the remote file name. Anything beyond that is truncated when the filename is stored in nonvolatile memory, but this truncation is not visible until the next time the system reboots.

### *Specifying End of Line Format*

By default, the bulk statistics application generates a DOS-compatible file that contains both a carriage return (CR) and line feed (LF) at the end of each line. The existence of a carriage return at the end of a line may cause formatting issues with some applications that do not ignore or remove carriage returns.

You can set up the system to remove the carriage return and leave only a line feed at the end of each line.

#### *bulkstats file-format endOfLine-LF*

- Use to strip the carriage return from the end of each line in the bulkstats file.
- Example

  host1(config)#**bulkstats file-format endOfLine-LF**

- Use the **no** version to return to the default, CR and LF.

## Managing Virtual Routers

Your router supports SNMP management of virtual routers. This support is based on an SNMP community string proxy to select particular instances of virtual routers. The entity MIB is used to model the physical container to the logical relationship of the virtual router implementation. See *Chapter 13, Configuring Virtual Routers*.

## Monitoring SNMP

To monitor the status of SNMP operations on your network, enter Privileged Exec mode. You can then establish a baseline and use the **show** commands to view statistics.

### *Establishing a Baseline*

SNMP statistics are stored in system counters. The only way to reset the system counters is to reboot the router. You can, however, establish a baseline for SNMP statistics by setting a group of reference counters to zero.

#### *baseline snmp*

- Use to establish a baseline for SNMP statistics.
- The system implements the baseline by reading and storing the statistics at the time the baseline is set and then subtracting this baseline whenever baseline-relative statistics are retrieved.
- To display statistics relative to the current baseline, use the **delta** keyword with SNMP **show** commands.
- SNMP operations (such as Get and Set) continue to use and report statistics from the system counters.
- See *Viewing SNMP Status* on page 205 for a sample display when you enter the **show snmp** command. If you establish a baseline and then enter **show snmp**, the statistics now have zero or low values.

- Example

```
host1#baseline snmp
host1#show snmp
Contact: Joe Administrator
Location: Network Lab, Bldg 3 Floor 1
2 SNMP packets input
    0 Bad SNMP version errors
    0 Unknown community name
    0 Illegal operation for community name supplied
    0 Encoding errors
    0 Number of requested variables
    0 Number of altered variables
    1 Get-request PDUs
    1 Get-next PDUs
    0 Set-request PDUs
    0 Unknown security models
    0 Unavailable contexts
2 SNMP packets out
    0 Too big errors (Maximum packet size 1500)
    1 No such name errors
    0 Bad values errors
    0 General errors
    2 Get-response PDUs
    0 SNMP trap PDUs
    0 Invalid Message Report PDUs
    0 Unknown PDU Handler Report PDUs
    0 Unknown Context Report PDUs
    0 Unsupported Security Level Report PDUs
    0 Not in time Window Report PDUs
    0 Unknown Username Report PDUs
    0 Unknown Engine ID Report PDUs
    0 Wrong Digest Report PDUs
    0 Decryption Error Report PDUs
```

- There is no **no** version.

## *Viewing SNMP Status*

To view SNMP status on your network, use the following **show** commands.

### *show snmp*

- Use to display all the information about SNMP status.
- To display statistics relative to the current baseline, use the **delta** keyword.
- Field descriptions
  - Contact—Router's contact person
  - Location—Router's location
  - SNMP packets input—Total number of SNMP packets received by the router
    - Bad SNMP version errors—Number of SNMP PDUs with a bad version number
    - Unknown community name—Number of SNMP PDUs that had an unrecognized community name

- Illegal operation for community name supplied—Number of access violations based on the configured privilege level for community strings
- Encoding errors—Number of AS number version 1 encoding and decoding errors
- Number of requested variables—Number of variable bindings processed by the SNMP agent
- Number of altered variables—Number of variable bindings processed successfully in SNMP **set** commands
- Get-request PDUs—Number of get-exact SNMP PDUs processed
- Get-next PDUs—Number of get-next SNMP PDUs processed
- Set-request PDUs—Number of set SNMP PDUs processed
- Unknown security models—Number of SNMP PDUs with unrecognized security
- Unavailable contexts—Number of SNMP proxy requests to unknown entities

- SNMP packets out—Total number of SNMP packets sent by the router
  - Too big errors—Number of processed PDUs that resulted in SNMP PDUs too large to encode
  - No such name errors—Number of requests that resulted in noSuchName errors. If interfaces configured on modules that do not support 64-bit counters are accessed, the system returns a noSuchName message.
  - Bad values errors—Number of requests that resulted in badValues errors
  - General errors—Number of general errors
  - Get-response PDUs—Number of requests that resulted in getResponse PDUs
  - SNMP trap PDUs—Number of SNMP trap PDUs generated by this agent
  - SNMP trap proxied—Number of traps generated by this agent that are sent via trap-proxy
  - Invalid Message Report PDUs—Number of packets received by the SNMP engine that were dropped because there were invalid or inconsistent components in the SNMP message
  - Unknown PDU Handler Report PDUs—Number of packets received by the SNMP engine that were dropped because the PDU in the packet could not be passed to an application responsible for handling the PDU type; for example, no SNMP application had registered for the proper combination of the context engine ID and PDU type
  - Unknown Context Report PDUs—Number of packets received by the SNMP engine that were dropped because the context contained in the message was unknown

- Unsupported Security Level Report PDUs—Number of packets received by the SNMP engine that were dropped because they requested a security level that was unknown to the SNMP engine or otherwise unavailable
- Not in time Window Report PDUs—Number of packets received by the SNMP engine that were dropped because they appeared outside the authoritative SNMP engine window
- Unknown Username Report PDUs—Number of packets received by the SNMP engine that were dropped because they referenced a user that was not known to the SNMP engine
- Unknown Engine ID Report PDUs—Number of packets received by the SNMP engine that were dropped because they referenced an snmpEngineID that was not known to the SNMP engine
- Wrong Digest Report PDUs—Number of packets received by the SNMP engine that were dropped because they did not contain the expected digest value
- Decryption Error Report PDUs—Number of packets received by the SNMP engine that were dropped because they could not be decrypted

- Example

```
host1#show snmp
Contact: Joe Administrator
Location: Network Lab, Bldg 3 Floor 1
538 SNMP packets input
    0 Bad SNMP version errors
    0 Unknown community name
    0 Illegal operation for community name supplied
    0 Encoding errors
    695 Number of requested variables
    0 Number of altered variables
    26 Get-request PDUs
    512 Get-next PDUs
    0 Set-request PDUs
    0 Unknown security models
    0 Unavailable contexts
538 SNMP packets out
    0 Too big errors (Maximum packet size 1500)
    10 No such name errors
    0 Bad values errors
    0 General errors
    538 Get-response PDUs
    0 SNMP trap PDUs
    0 Invalid Message Report PDUs
    0 Unknown PDU Handler Report PDUs
    0 Unknown Context Report PDUs
    0 Unsupported Security Level Report PDUs
    0 Not in time Window Report PDUs
    0 Unknown Username Report PDUs
    0 Unknown Engine ID Report PDUs
    0 Wrong Digest Report PDUs
    0 Decryption Error Report PDUs
```

### *show snmp access*

- Use to display information about the groups you configured.
- Field descriptions
  - Group Name—Name of the group
  - Model—Security model; for example, user-based security model (USM)
  - Level—Method for authentication and privacy
    - none—No authentication and no privacy
    - auth—Authentication only
    - priv—Authentication and privacy
  - Read—Name of the view for read access
  - Write—Name of the view for write access
  - Notify—Name of the view for notification
  - Storage—SNMP storage type, volatile or nonvolatile
- Example

```
host1#show snmp access
Group Name          Model  Level  Read         Write        Notify
------------------  -----  -----  -----------  -----------  ---------
admin               usm    priv   everything   everything   everything
mirror              usm    priv   mirrorAdmin  mirrorAdmin  mirrorAdmin
public              usm    none   user         none         none
private             usm    auth   user         user         user
```

### *show snmp community*

- Use to display information about the SNMP communities.
- Field descriptions
  - Community—Name of the community and the associated virtual router
  - View—Name of the view
  - Priv—Access privilege for the view
    - ro—Read-only access
    - rw—Read-write access
    - admin—All privileges
  - AccList—Number of access lists associated with this community
- Example

```
host1#show snmp community

Community                    View           Priv  AccList
-------------------------------------------  ----  -------
admin@default                everything     rw          0
private@default              user           rw          0
public@default               user           ro          0
```

***show snmp group***

- Use to display the list of available groups. Detailed information is available through the **show snmp access** command.
- Field descriptions
  - groupName—Name of the group
  - *securityModel*—SNMP security model
    - v1—SNMPv1
    - v2c—SNMPv2c
    - usm—SNMPv
  - authenticationLevel—Method for authentication and privacy
    - none—No authentication and no privacy
    - auth—Authentication only
    - priv—Authentication and privacy
  - readView—Name of the view for read access
  - writeView—Name of the view for write access
  - notifyView—Name of the view for notification
  - storageType—SNMP storage type
    - volatile—Loses contents when power is lost
    - nonVolatile—Does not lose contents when power is lost
- Example

```
host1#show snmp group
Group Name                                Storage Type
----------------------------------------  ---------------
group1                                    Volatile
group2                                    NonVolatile
admin                                     Permanent
mirror                                    Permanent
public                                    Permanent
private                                   Permanent
```

***show snmp notificationLog***

- Use to display the configuration of the SNMP notification log.
- Field descriptions
  - Global Age Out Value—Ageout for traps in the notification log tables
  - Global Entry Limit Value—Maximum number of notifications kept in all notification log tables
- Example

```
host1#show snmp notificationLog
Global Age Out Value:      1440 minutes
Global Entry Limit Value : 500
No notification log name information is available
```

***show snmp trap***

- Use to display configuration information about SNMP traps and trap destinations.
- Field descriptions
  - Enabled Categories—Trap categories that are enabled on the router
  - SNMP authentication failure trap—Enabled or disabled
  - Trap Source—Interface whose IP address is used as the source address for all SNMP traps
  - Trap Source Address—IP address used as the source address for all SNMP traps
  - Trap Proxy—Enabled or disabled
  - Global Trap Severity Level—Global severity level filter; if a trap does not meet this severity level, it is discarded
  - Address—IP address of the trap recipient
  - Security String—Name of the SNMP community
  - Ver—SNMP version (v1 or v2) of the SNMP trap packet
  - Port—UDP port on which the trap recipient accepts traps
  - Trap Categories—Types of traps that the trap recipient can receive
  - TrapSeverityFilter—Severity level filter for this SNMP host
  - Ping TimeOut—Configured ping timeout in minutes
  - Maximum QueueSize—Maximum number of traps to be kept in the trap queue
  - Queue DrainRate—Maximum number of traps per second to be sent to the host
  - Queue Full discard method—Method used to discard traps when the queue is full:
    - dropFirstIn—Oldest trap in the queue is dropped
    - dropLastIn—Most recent trap is dropped
- Example

```
host1#show snmp trap

Enabled Categories: Ping, TraceRoute
SNMP authentication failure trap is disabled
Trap Source: FastEthernet 0/0, Trap Source Address:10.10.5.61
Trap Proxy: disabled
Global Trap Severity Level: 6 - informational

Address          Security String                   Ver  Port   Trap Categories
---------------  --------------------------------  ---  -----  ----------------
10.10.0.200       private                           v2c 162
SnmpLinkInvEnvBstFxfBgpLogCliPingOspfTraceDvmrpDvmrpUniAdrPAtmPingVrrpSonetNtp


Address         TrapSeverityFilter  Ping    Maximum    Queue    Queue Full
                                   TimeOut QueueSize DrainRate discrd methd
--------------- ------------------ ------- --------- --------- -------------
10.10.0.200      5 - notice         1       32        0         dropLastIn
```

***show snmp trap statistics***

- Use to display statistics for all SNMP traps on the virtual router, as well as statistics for each SNMP host configured on the virtual router.
- Field descriptions
  - Trap request(s)—Number of local traps requested
  - Proxy trap request(s)—Number of proxy traps requested
  - Trap(s) discarded—Total number of traps discarded
    - No system memory—Traps discarded because there was not enough system memory
    - No queue resources—Traps discarded because there were no queue resources available
    - SNMP agent disabled—Traps discarded because the SNMP agent was disabled
    - Global trap category disabled—Traps discarded because they were filtered by the **snmp enable trap** command
    - Global minimum severity level—Traps discarded because they did not match the severity level set with the **snmp enable traps trapfilters** command.
  - Trap(s) out—Total number of traps sent by the virtual router
  - Trap(s) proxied—Total number of traps proxied by the virtual router
  - Address—IP address of the host
  - TrapsDiscarded Severity/Category—Severity level and category of the discarded traps
  - TrapsDiscrded bad encoding—Traps discarded because of bad encoding
  - TrapsDiscrded Queue Full—Traps discarded because the queue was full
  - TrapsDiscrded NoHostRespons—Traps discarded because the host did not respond to pings sent to the host
  - Trap PDUs sentOut—Number of trap PDUs sent by this host

```
host1#show snmp trap statistics
Trap request(s):3112
Proxy trap request(s):0
Trap(s) discarded:4
    No system memory:0
    No queue resources:0
    SNMP agent disabled:0
    Global trap category disabled:4
    Global minimum severity level:0
Trap(s) out:3108
Trap(s) proxied:0
Address         TrapsDiscarded    TrapsDiscrded TrapsDiscrded TrapsDiscrded
                Severity/Category bad encoding  Queue Full    NoHostRespons
--------------- ----------------- ------------- ------------- -------------
1.1.1.1         1081              0             511           32
10.10.132.137   0                 0             0             0
```

```
Address         Trap PDUs
                sentOut
--------------- ---------
1.1.1.1         536
10.10.132.137   3108
```

### *show snmp user*

- Use to display information about users.
- Field descriptions
  - User—Name of the user
  - Auth—Authorization protocol for this user
    - no—No authorization protocol
    - md5—HMAC-MD5-96 authorization protocol
    - sha—HMAC-SHA-96 authorization protocol
  - Priv—Privacy protocol for this user
    - no—No privacy protocol
    - des—DES encryption algorithm for privacy
  - Group—Name of the group to which the user belongs
- Example SNMPv3 display.

```
host1#show snmp user

User                    Auth  Priv  Group
----------------------- ----  ----  ------------------
josie                   md5   des   admin
nightfly                md5   no    private
steelydan               no    no    public
```

### *show snmp view*

- Use to display information about the views you created.
- Field descriptions
  - View Name—Name of the view
  - View Type—Access privilege for the view
    - included—Specified object identifier (OID) trees are available in this view
    - excluded—Specified OID trees are not available in this view
  - Oid Tree—OID of the AS number version 1 subtree
  - Storage—SNMP storage type, volatile or nonvolatile
- Example

```
host1#show snmp view

View Name       View Type  Oid Tree
--------------  ---------  --------------------------
user            included   1.3.6.1.
user            excluded   1.3.6.1.4.1.4874.2.2.16.
user            excluded   1.3.6.1.6.3.11.
user            excluded   1.3.6.1.6.3.12.
```

```
user            excluded   1.3.6.1.6.3.13.
user            excluded   1.3.6.1.6.3.14.
user            excluded   1.3.6.1.6.3.15.
user            excluded   1.3.6.1.6.3.16.
user            excluded   1.3.6.1.6.3.18.
nothing         excluded   1.3.6.1.
everything      included   1.3.6.1.
everything      excluded   1.3.6.1.4.1.4874.2.2.77.
mirrorAdmin     included   1.3.6.1.4.1.4874.2.2.77.
```

### *Output Filtering*

You can use the output filtering feature of the **show** commands to include or exclude lines of output based on a text string you specify. See *Chapter 2, Command-Line Interface*, for details.