

# 4

## Configuring SNMP

This chapter provides information for configuring Simple Network Management Protocol (SNMP) on your ERX system.

Topic	Page
Overview	4-1
References	4-11
Before You Configure SNMP	4-11
SNMP Configuration Tasks	4-12
Configuring Traps	4-19
Collecting Bulk Statistics	4-27
Using the Bulk Statistics Formatter	4-44
Managing Virtual Routers	4-45
Monitoring SNMP	4-46

### Overview

---

SNMP is a protocol that manages network devices, such as your ERX system. 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 system’s status to the client. In this instance, the information is unsolicited.

### Terminology

Table 4-1 provides definitions for the basic SNMP terms.

**Table 4-1** 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 system
MIB	Management Information Base; a collection of managed objects residing in a virtual information store

**Table 4-1** SNMP terminology (continued)

Term	Meaning
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
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 system
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 ASN.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



**Note:** Your system allows you to disable the management interface via SNMP. If you disable the management interface, you can no longer access the system via SNMP.

### *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 system. 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 system 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 ERX 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 system, slot, and subsystem. The extensions also enable booting via the factory defaults, the running configuration, or a configuration (.cnf) file.

#### Accessing Supported SNMP MIBs

For complete information on the SNMP MIBs supported by your system, see the *System Software* CD, shipped with your system. In the MIBs folder you will find information on all supported standard and Juniper Networks ERX 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 1905)
- SNMPv3 (compliant with RFCs 2570–2575)

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 the following:

- Security for messages
- Explicit access control

## *Security Features*

As users transfer more sensitive information, such as billing details, via 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 via 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, SNMPv1/v2c provide only password protection, via 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. Three predefined views are available for each group:
  - > everything – Includes all MIBs associated with the system
  - > user – Includes all MIBs associated with the system, except standard and enterprise MIBs used to configure SNMP operation
  - > nothing – Excludes all MIBs

- User – An individual who requires access to the system. The system may provide authentication and privacy for the user via SNMPv3. Each user is associated with a group.
- Group – A set of users with the same access privileges to the system. Three predefined groups are available: admin, public, and private. Table 4-2 shows the security levels and views associated with these groups.

**Table 4-2** Relationship between groups, security levels, and views

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

### Virtual Routers

All SNMP-related CLI commands operate in the context of the 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 system software allows you to configure multiple virtual routers. Each virtual router has its own SNMP server. At system initialization, SNMP creates a server for each existing virtual router.

In cases where router-specific data is required, the requestor can direct a request to a particular server for a virtual router via 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 router 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 ERX system implementation of SNMPv3 communicates with virtual routers by assigning each proxy agent an *SNMP engineID*. This difference is unimportant to users of the CLI. However, if you use other SNMPv3 applications to manage the system, refer to the following section.

### 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 *SnmEngineID*, a hexadecimal number 15 octets long. Table 4-3 shows the structure of the *SnmEngineID*.

**Table 4-3** Structure of the SnmpEngineID

Octet Assignment	Description
1 – 4	ERX system SNMP management private enterprise number
5	Indicates that octets 6–15 contain information determined by the ERX system
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 *contextEngineID* and *contextName* for the SNMP engine. When the system receives a PDU, it examines the *contextEngineID* and *contextName*, and forwards the request to the corresponding virtual router.

- The *contextEngineID* is the same as the *SnmEngineID*.
- 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 via the Unisphere-Data-ROUTER-MIB from the *usdRouterContextName* object in the *usdRouterTable*, which is indexed by the 32-bit router index (*usdRouterIndex*).

**Examples** The following table shows examples of the ERX system SNMP engine objects that are associated with the default virtual router.

Object	Value
SnmEngineID	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 4-4.

**Table 4-4** 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 4-5.

**Table 4-5** 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 4-6.

**Table 4-6** 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 <i>after</i> 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.
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.)

## 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 1905 – Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2) (January 1996)
- RFC 2570 – Introduction to Version 3 of the Internet-standard Network Management Framework (April 1999)
- RFC 2571 – An Architecture for Describing SNMP Management Frameworks (April 1999)
- RFC 2572 – Message Processing and Dispatching for the Simple Network Management Protocol (SNMP) (April 1999)
- RFC 2573 – SNMPv3 Applications (April 1999)
- RFC 2574 – User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3) (April 1999)
- RFC 2575 – View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP) (April 1999)
- RFC 3014 – Notification Log MIB (November 2000)

## Before You Configure SNMP

---

Before you configure SNMP, make sure that at least one IP address is configured on your system. See *ERX Installation and User Guide, Chapter 5, Accessing the ERX System*.

You should also 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 auth sha fred-password  
priv des password group user
```

- 3 (Optional) Set the server parameters—contact and 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 system 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
```

You can also set up SNMP traps and set up the system to collect bulk statistics. See *Configuring Traps* and *Collecting Bulk Statistics* later in this chapter.

## 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.
- Example

```
host1(config)#snmp-server community "boston" view everything
                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.

### *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 system 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 system 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 ERX system 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 to the ERX system proprietary method.
- Example
 

```
host1(config)#snmp 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` and the `ifStackTable`, 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` and the `ifStackTable`.

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

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

```
host1(config)#snmp 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 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 and the ifStackTable.
- Example

```
host1(config)#snmp-server interfaces compress atmAal5
```
- Use the **no** version to add interface sublayers to the ifTable and the ifStackTable.

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

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

```
host1(config)#snmp-server interface compress-restriction  
ifadminstatusdown
```

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

### 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 ERX system implementation of SNMP derives index numbers in 32-bit values that are unique on a given system. 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 system uses a numbering scheme based on RFC 1573. For compatibility with RFC 1213, you can set up the system 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 system 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 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 interfaces***

- Use to display a list of interface types that are compressed in the interface tables and the interface numbering method configured on the system.
- 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 system: RFC1213, RFC1573
  - › 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: industry-common, proprietary

```
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 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 system (the server). Each entry in the table contains:

- IP address of the trap destination
- Community name (v1 or v2c) or user name (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 system supports the following trap categories:

- addrPool – local address pool
- atmPing – ERX system proprietary ATM ping
- bgp – BGP state change
- bulkstats – bulk statistics file full and nearly full
- cliSecurityAlert – security alert
- dvmrp – DVMRP
- dvmrpUni – ERX system proprietary DVMRP
- environment – power, temperature, fan, and memory utilization
- fileXfer – file transfer status change
- inventory – system inventory and status
- link – SNMP linkUp and linkDown
- log – system log capacity

- ntp – ERX system proprietary traps
- ospf – OSPF
- ping – ping operation (in disman remops MIB)
- radius – RADIUS servers fail to respond to accounting and authentication requests
- snmp – SNMP coldstart, warmstart, authenticationFailure; the trap option. The **snmp-server enable traps snmp authentication** command allows customized treatment for SNMP authentication failure traps
- traceroute – traceroute operation (in disman remops MIB)

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 system provides a method of filtering traps according to severity. Table 4-7 describes the supported severity levels.

**Table 4-7** 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 checks 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 checks 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* earlier in this chapter.
- You can filter traps according to the trap severity levels described in Table 4-7.
- 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* earlier in this chapter.
- You can filter traps according to the trap severity levels described in Table 4-7.
- 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.

### *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 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 system.

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 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. The ERX system also records all SNMP traps that are generated from the time the power switch is turned on.

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 ERX system. Clients can use the index to detect missing traps.

To configure trap notification logs:

- 1 Enable the snmpTrap log to severity level info.

```
host1(config)#log severity info snmpTrap
```
- 2 Configure the notification log.

```
host1(config)#snmp-server notificationlog log 10.10.4.4  
adminStatus includeVarbinds
```
- 3 (Optional) Specify when the notification log ages out.

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

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

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

### log severity



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

**Note:** For more information on this command, see *ERX System Basics Configuration Guide, Chapter 11, Logging System Events*.

- 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 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.

### 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 system. 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.

### *Recovering Lost Traps*

SNMP traps can be lost during startup of the ERX system 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 ERX system 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, you should 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).
- 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.

## Collecting Bulk Statistics

---

The system offers an efficient data collection and transfer facility for accounting applications. The ERX system 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.
- It is expensive to process SNMP PDUs because objects and tables need to be sorted in lexicographic order.

The system 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 system sends collected statistics via 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 BER (basic encoding rules) encoding choice is not supported.

### Configuring Collectors and Receivers

To configure the system to collect statistics:

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

See *Using the copy Command* 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 7
```

- 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
```



**Note:** Bulk statistics 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.
- 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, 3670016 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 bulkstats 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* earlier in this chapter.
- 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 type on which you want to collect statistics.
- The supported interface types are:
  - › ATM
  - › ATM 1483
  - › Ethernet
  - › Frame Relay
  - › Frame Relay subinterface
  - › Cisco HDLC
  - › IP
  - › PPP
- Example  

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

- If you define more than one collector, you must specify a unique collector index, in the range 1–65535.
- You can collect statistics on interfaces for the FE-2 module and the Gigabit Ethernet module. You cannot collect statistics on the SRP Ethernet interface.
- Example
 

```
host1(config)#bulkstats interface-type ethernet collector 2
```
- 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 FTP file transfer supports only anonymous transfers to remote servers. Other user names and passwords are not supported.
- The receivers must appear in the FTP host table (see *Using the copy Command in Chapter 5, Managing the System*). 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 run time 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.

### *Monitoring Collection Statistics*

To view the parameters the system 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
  - › Interface Description Setting – method used to encode the ifDescr object: common, proprietary
  - › 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 against 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
    - › 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 – NMS 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/operational error
  - › Index – bulk statistics collector index
  - › Primary-Receiver – index of the primary receiver to which the system transfers data
  - › 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
- › Index – bulk statistics collector index
- › 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
  - › 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/operational error
  - › Index – index number of the schema
  - › Subtree List – type(s) 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 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
- › Receiver Information:
  - › 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

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	CollectorIndex	State
1	ifStats	1	active
2	ifStack	2	active

Index Subtree List

1	ifInOctets; ifOutUcastPkts; ifOutPolicedOctets
2	N/A

## 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

**show bulkstats collector description**

- Use to display information on 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 on 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 on 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 on 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 on 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 on the bulk statistics receiver's remote file configuration.
- 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 system
  - › CollectorFileFullTraps – number of file full events posted to the SNMP agent on this system
  - › 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
    - › Index – bulk statistics collector index

- › 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

- 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
```

### show bulkstats traps

- Use to display information on 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
  
```

### 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 system. The bulk statistics application supports four schema configurations: *if-stack*, *if-stats*, *policy*, and *system*. Table 4-8 shows the type of data each schema retrieves.

**Table 4-8** Data retrieved according to schema

Schema	Retrieves . . .
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 system.
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.

## if-stats Objects

Table 4-9 presents if-stats objects you can configure using the **bulkstats schema subtree** command.

**Table 4-9** Schema ifStats objects

Object	Definition
usdAcctngIfInBroadcastPkts	Broadcast packets received
usdAcctngIfInOctets	Octets received; support 64-bit counters
usdAcctngIfInUcastPkts	Unicast packets received
usdAcctngIfInDiscards	Packets received and discarded
usdAcctngIfInErrors	Packets received with errors
usdAcctngIfInMulticastPkts	Multicast packets received
usdAcctngIfInUnknownProtos	Packets received with unknown protocols
usdAcctngIfOutBroadcastPkts	Broadcast packets sent
usdAcctngIfOutOctets	Octets sent; support 64-bit counters
usdAcctngIfOutUcastPkts	Unicast packets sent
usdAcctngIfOutDiscards	Packets sent and discarded
usdAcctngIfOutErrors	Packets sent with errors
usdAcctngIfOutMulticastPkts	Multicast packets sent
usdAcctngIfCorrelator	Customer correlation: <ul style="list-style-type: none"> <li>• FR = DLCI</li> <li>• ATM = VPI, VCI</li> <li>• IP = RouterName</li> <li>• Everything else = not used</li> </ul>
usdAcctngIfInPolicedOctets	Octets dropped due to ingress policy; support 64-bit counters.
usdAcctngIfInPolicedPkts	Packets dropped due to ingress policy
usdAcctngIfInSpoofedPkts	Packets dropped due to invalid source address
usdAcctngIfOutPolicedOctets	Octets dropped due to egress policy; support 64-bit counters
usdAcctngIfOutSpoofedPkts	Packets dropped due to invalid source address
usdAcctngIfOutSchedulerDropPkts	Scheduler packets dropped
usdAcctngIfOutSchedulerOctets	Scheduler octets dropped
usdAcctngIfLowerInterface	The ifIndex of the lower interface



**Note:** All the schema if-stats objects in Table 4-9 apply to both layer 2 and layer 3 interfaces, except *usdAcctngSpoofedPkts*, which is specific to layer 3.

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.

### **bulkstats schema**

- Use to create the schema for collecting bulk statistics.
- Example
- Use the **no** version to delete the specified schema.

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



**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.
  - › **system** – retrieves global system and per-module statistics and information
- Example
- Use the **no** version to delete the specified schema.

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

### **bulkstats schema subtree policy**

- 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, or output policies
  - › **policy-subtreelist** – collects statistics based on color-coded tags applied by a policy
- You create policies using the **policy-list** command. See *ERX Policy and QoS Configuration Guide, Chapter 1, Configuring Policy Management*.
- Example
- Use the **no** version to delete the specified schema.

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

## 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: 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
  - › Index – index number of the schema
  - › 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
```

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

## 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 system 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 run time 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 re-add the bulk statistics receiver.
  - You can use up to 128 characters for the remoteFileName. 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 bulkstats file contains a CR and LF at the end of each line. You can set up the system to remove the CR and leave only an LF 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 system 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 10, 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 system. 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* later in this chapter 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 on 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 ASN.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
  - 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 pduType; for example, no SNMP application had registered for the proper combination of the contextEngineID and pduType
  - 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 of 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

- Example

```
host1#show snmp access
```

Group Name	Model	Level	Read	Write	Notify
admin	usm	priv	everything	everything	everything
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 notificationLog**

- Use to display the configuration of the SNMP notification log.
- Field descriptions
  - › Global Age Out Value – age out 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 on 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 discrd methd – method used to discard traps when the queue is full:
  - dropFirstIn – oldest trap in the queue is dropped
  - dropLastIn – most recent trap is dropped

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

**show snmp trapStats**

- 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
  - › Traps received – total number of traps received
    - Local traps submitted – number of local traps submitted
    - Proxy traps submitted – number of proxy traps submitted
  - › Traps discarded – total number of traps discarded
    - Lack of 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 being disabled – traps discarded because the SNMP agent was disabled
    - Global category control – 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.
  - › Traps out – total number of packets sent by the virtual router
  - › Traps 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 trapstat
```

```
3112 Traps received
    3112 Local traps submitted
    0 Prxoy traps submitted
4 Traps discarded
    0 Lack of system memory
    0 No queue resources
    0 SNMP agent being disabled
    4 Global category control
    0 Global minimum severity level
```

```
3108 Traps out
0 Traps proxied
```

Address	TrapsDiscarded Severity/Category	TrapsDiscrded bad encoding	TrapsDiscrded Queue Full	TrapsDiscrded 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
- The following example is an SNMPv3 display.
- Example

```
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 ASN.1 subtree
- Example

```
host1#show snmp view
```

View Name	View Type	Oid Tree
-----	-----	-----
everything	included	1.3.6.1.

```
user          included  1.3.6.1.  
user          excluded 1.3.6.1.4.1.2773.2.16.  
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.
```

### *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.