# Configuring Simple Network Management Protocol

Simple Network Management Protocol (SNMP) is an application-layer protocol that provides a message format for communication between SNMP managers and agents. SNMP provides a standardized framework and a common language used for the monitoring and management of devices in a network.

This module describes the tasks you need to implement SNMP on your Cisco IOS XR network.


- Prerequisites for Implementing SNMP, on page 1
- Restrictions for SNMP use on Cisco IOS XR Software, on page 1
- Information about Implementing SNMP, on page 2
- Session MIB support on subscriber sessions , on page 7
- How to Implement SNMP on Cisco IOS XR Software, on page 9


## Prerequisites for Implementing SNMP

You must be in a user group associated with a task group that includes the proper task IDs. The command reference guides include the task IDs required for each command. If you suspect user group assignment is preventing you from using a command, contact your AAA administrator for assistance.

## Restrictions for SNMP use on Cisco IOS XR Software

SNMP outputs are only 32-bits wide and therefore cannot display any information greater than $2^{32}$. $2^{32}$ is equal to 4.29 Gigabits.

**Note** A 10 Gigabit interface is greater than $2^{32}$, so if you are trying to display speed information regarding the interface, you might see concatenated results.

To display correct speed of an interface greater than 10 Gigabit, ifHighSpeed can be used.

The recommended maximum number of object identifiers (OIDs) that can be accommodated in a single SNMP request is 75. A request with more than 75 OIDs can result in SNMP requests being dropped with SNMP polling timeout.

# Information about Implementing SNMP

To implement SNMP, you need to understand the concepts described in this section.

## SNMP Functional Overview

The SNMP framework consists of three parts:

- SNMP manager
- SNMP agent
- Management Information Base (MIB)

### SNMP Manager

The SNMP manager is the system used to control and monitor the activities of network hosts using SNMP. The most common managing system is called a *network management system* (NMS). The term NMS can be applied to either a dedicated device used for network management, or the applications used on such a device. A variety of network management applications are available for use with SNMP. These features range from simple command-line applications to feature-rich graphical user interfaces (such as the CiscoWorks 2000 line of products).

### SNMP Agent

The SNMP agent is the software component within the managed device that maintains the data for the device and reports these data, as needed, to managing systems. The agent and MIB reside on the router. To enable the SNMP agent, you must define the relationship between the manager and the agent.

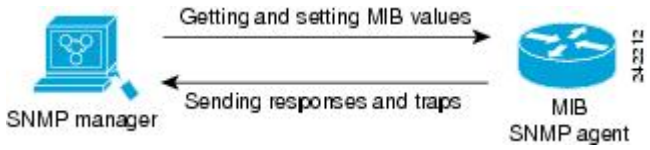
### MIB

The *Management Information Base* (MIB) is a virtual information storage area for network management information, which consists of collections of managed objects. Within the MIB there are collections of related objects, defined in MIB modules. MIB modules are written in the SNMP MIB module language, as defined in STD 58, RFC 2578, RFC 2579, and RFC 2580. Note that individual MIB modules are also referred to as MIBs; for example, the Interfaces Group MIB (IF-MIB) is a MIB module within the MIB on your system.

The SNMP agent contains MIB variables whose values the SNMP manager can request or change through Get or Set operations. A manager can get a value from an agent or store a value into that agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to manager requests to get or set data.

This figure illustrates the communications relationship between the SNMP manager and agent. A manager can send the agent requests to get and set MIB values. The agent can respond to these requests. Independent of this interaction, the agent can send unsolicited notifications (traps) to the manager to notify the manager of network conditions.

***Figure 1: Communication Between an SNMP Agent and Manager***



### IP-MIB Support

RFC4293 IP-MIB was specifically designed to provide IPv4 and IPv6 statistics individually. The **ipIfStatsTable** defined in RFC 4293, lists the interface specific statistics. IPv6 statistics support in ipIfStatsTable was added earlier but, IOS-XR implementation of IP-MIB did not support IPv4 statistics as per RFC4293 in earlier releases.

From Release 6.3.2 onwards, IOS-XR implementation of IP-MIB supports IPv4 statistics as per RFC4293. This will enable you to collect the IPV4 and IPv6 statistics separately for each interface. The **ipIfStatsTable** is indexed by two **sub-ids address type (IPv4 or IPv6)** and the **interface ifindex[1]**. The implementation of IP-MIB support for IPv4 and IPv6 is separated from Release 6.3.2 for better readability and maintainability.

The list of OIDs added to the **ipIfStatsTable** for IPv4 statistics are:

- ipIfStatsInReceives
- ipIfStatsHCInReceives
- ipIfStatsInOctets
- ipIfStatsHCInOctets
- ipIfStatsOutTransmits
- ipIfStatsHCOutTransmits
- ipIfStatsOutOctets
- ipIfStatsHCOutOctets
- ipIfStatsDiscontinuityTime

For more information on the list of new OIDs added for iPv4 statistics, see SNMP OID Navigator.

# SNMP Versions

Cisco IOS XR software supports the following versions of SNMP:

- Simple Network Management Protocol Version 1 (SNMPv1)
- Simple Network Management Protocol Version 2c (SNMPv2c)
- Simple Network Management Protocol Version 3 (SNMPv3)

Both SNMPv1 and SNMPv2c use a community-based form of security. The community of managers able to access the agent MIB is defined by an IP address access control list and password.

SNMPv2c support includes a bulk retrieval mechanism and more detailed error message reporting to management stations. The bulk retrieval mechanism supports the retrieval of tables and large quantities of information, minimizing the number of round-trips required. The SNMPv2c improved error handling support includes expanded error codes that distinguish different kinds of error conditions; these conditions are reported through a single error code in SNMPv1. Error return codes now report the error type. Three kinds of exceptions are also reported: no such object exceptions, no such instance exceptions, and end of MIB view exceptions.

SNMPv3 is a security model. A *security model* is an authentication strategy that is set up for a user and the group in which the user resides. A *security level* is the permitted level of security within a security model. A combination of a security model and a security level will determine which security mechanism is employed

when an SNMP packet is handled. See Security Models and Levels for SNMPv1, v2, v3, on page 4 for a list of security levels available in SNMPv3. The SNMPv3 feature supports RFCs 3411 to 3418.

You must configure the SNMP agent to use the version of SNMP supported by the management station. An agent can communicate with multiple managers; for this reason, you can configure the Cisco IOS-XR software to support communications with one management station using the SNMPv1 protocol, one using the SNMPv2c protocol, and another using SMNPv3.

## Comparison of SNMPv1, v2c, and v3

SNMP v1, v2c, and v3 all support the following operations:

- get-request—Retrieves a value from a specific variable.
- get-next-request—Retrieves the value following the named variable; this operation is often used to retrieve variables from within a table. With this operation, an SNMP manager does not need to know the exact variable name. The SNMP manager searches sequentially to find the needed variable from within the MIB.
- get-response—Operation that replies to a get-request, get-next-request, and set-request sent by an NMS.
- set-request—Operation that stores a value in a specific variable.
- trap—Unsolicited message sent by an SNMP agent to an SNMP manager when some event has occurred.

This table identifies other key SNMP features supported by the SNMP v1, v2c, and v3.

***Table 1: SNMPv1, v2c, and v3 Feature Support***

| Feature | SNMP v1 | SNMP v2c | SNMP v3 |
|---|---|---|---|
| Get-Bulk Operation | No | Yes | Yes |
| Inform Operation | No | Yes (No on the Cisco IOS XR software) | Yes (No on the Cisco IOS XR software) |
| 64 Bit Counter | No | Yes | Yes |
| Textual Conventions | No | Yes | Yes |
| Authentication | No | No | Yes |
| Privacy (Encryption) | No | No | Yes |
| Authorization and Access Controls (Views) | No | No | Yes |

## Security Models and Levels for SNMPv1, v2, v3

The security level determines if an SNMP message needs to be protected from disclosure and if the message needs to be authenticated. The various security levels that exist within a security model are as follows:

- noAuthNoPriv—Security level that does not provide authentication or encryption.
- authNoPriv—Security level that provides authentication but does not provide encryption.

- authPriv—Security level that provides both authentication and encryption.

Three security models are available: SNMPv1, SNMPv2c, and SNMPv3. The security model combined with the security level determine the security mechanism applied when the SNMP message is processed.

The below table identifies what the combinations of security models and levels mean.

***Table 2: SNMP Security Models and Levels***

| Model | Level | Authentication | Encryption | What Happens |
|---|---|---|---|---|
| v1 | noAuthNoPriv | Community string | No | Uses a community string match for authentication. |
| v2c | noAuthNoPriv | Community string | No | Uses a community string match for authentication. |
| v3 | noAuthNoPriv | Username | No | Uses a username match for authentication. |
| v3 | authNoPriv | HMAC-MD5 or HMAC-SHA | No | Provides authentication based on the HMAC[1]-MD5[2] algorithm or the HMAC-SHA[3]. |
| v3 | authPriv | HMAC-MD5 or HMAC-SHA | DES | Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides DES[4] 56-bit encryption in addition to authentication based on the CBC[5] DES (DES-56) standard. |
| v3 | authPriv | HMAC-MD5 or HMAC-SHA | 3DES | Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides 168-bit 3DES[6] level of encryption. |
| v3 | authPriv | HMAC-MD5 or HMAC-SHA | AES | Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides 128-bit AES[7] level of encryption. |

1 Hash-Based Message Authentication Code
2 Message Digest 5
3 Secure Hash Algorithm
4 Data Encryption Standard
5 Cipher Block Chaining
6 Triple Data Encryption Standard
7 Advanced Encryption Standard

Use of 3DES and AES encryption standards requires that the security package (k9sec) be installed. For information on installing software packages, see *Upgrading and Managing Cisco IOS XR Software*.

# SNMPv3 Benefits

SNMPv3 provides secure access to devices by providing authentication, encryption and access control. These added security benefits secure SNMP against the following security threats:

- Masquerade—The threat that an SNMP user may assume the identity of another SNMP user to perform management operations for which that SNMP user does not have authorization.

- Message stream modification—The threat that messages may be maliciously reordered, delayed, or replayed (to an extent that is greater than can occur through the natural operation of a subnetwork service) to cause SNMP to perform unauthorized management operations.
- Disclosure—The threat that exchanges between SNMP engines could be eavesdropped. Protecting against this threat may be required as a matter of local policy.

In addition, SNMPv3 provides access control over protocol operations on SNMP managed objects.

# SNMPv3 Costs

SNMPv3 authentication and encryption contribute to a slight increase in the response time when SNMP operations on MIB objects are performed. This cost is far outweighed by the security advantages provided by SNMPv3.

This table shows the order of response time (from least to greatest) for the various security model and security level combinations.

***Table 3: Order of Response Times from Least to Greatest***

| Security Model | Security Level |
|---|---|
| SNMPv2c | noAuthNoPriv |
| SNMPv3 | noAuthNoPriv |
| SNMPv3 | authNoPriv |
| SNMPv3 | authPriv |

## User-Based Security Model

SNMPv3 User-Based Security Model (USM) refers to SNMP message-level security and offers the following services:

- Message integrity—Ensures that messages have not been altered or destroyed in an unauthorized manner and that data sequences have not been altered to an extent greater than can occur nonmaliciously.
- Message origin authentication—Ensures that the claimed identity of the user on whose behalf received data was originated is confirmed.
- Message confidentiality—Ensures that information is not made available or disclosed to unauthorized individuals, entities, or processes.

SNMPv3 authorizes management operations only by configured users and encrypts SNMP messages.

USM uses two authentication protocols:

- HMAC-MD5-96 authentication protocol
- HMAC-SHA-96 authentication protocol

USM uses Cipher Block Chaining (CBC)-DES (DES-56) as the privacy protocol for message encryption.

## View-Based Access Control Model

The View-Based Access Control Model (VACM) enables SNMP users to control access to SNMP managed objects by supplying read, write, or notify access to SNMP objects. It prevents access to objects restricted by

views. These access policies can be set when user groups are configured with the **snmp-server group** command.

### MIB Views

For security reasons, it is often valuable to be able to restrict the access rights of some groups to only a subset of the management information within the management domain. To provide this capability, access to a management object is controlled through MIB views, which contain the set of managed object types (and, optionally, the specific instances of object types) that can be viewed.

### Access Policy

Access policy determines the access rights of a group. The three types of access rights are as follows:

- read-view access—The set of object instances authorized for the group when objects are read.
- write-view access—The set of object instances authorized for the group when objects are written.
- notify-view access—The set of object instances authorized for the group when objects are sent in a notification.

## IP Precedence and DSCP Support for SNMP

SNMP IP Precedence and differentiated services code point (DSCP) support delivers QoS specifically for SNMP traffic. You can change the priority setting so that SNMP traffic generated in a router is assigned a specific QoS class. The IP Precedence or IP DSCP code point value is used to determine how packets are handled in weighted random early detection (WRED).

After the IP Precedence or DSCP is set for the SNMP traffic generated in a router, different QoS classes cannot be assigned to different types of SNMP traffic in that router.

The IP Precedence value is the first three bits in the type of service (ToS) byte of an IP header. The IP DSCP code point value is the first six bits of the differentiate services (DiffServ Field) byte. You can configure up to eight different IP Precedence markings or 64 different IP DSCP markings.

# Session MIB support on subscriber sessions

SNMP monitoring requires information about subscribers of all types. The CISCO-SUBSCRIBER-SESSION-MIB is defined to model per-subscriber data as well as aggregate subscriber (PPPoE) data. It is required to support notifications (traps) for aggregate session counts crossing configured thresholds. Generic MIB Data Collector Manager (DCM) support for CISCO-SUBSCRIBER-SESSION-MIB, helps faster data collection and also better handling of parallel data.

## SNMP Notifications

A key feature of SNMP is the ability to generate notifications from an SNMP agent. These notifications do not require that requests be sent from the SNMP manager. On Cisco IOS XR software, unsolicited (asynchronous) notifications can be generated only as *traps*. Traps are messages alerting the SNMP manager to a condition on the network. Notifications can indicate improper user authentication, restarts, the closing of a connection, loss of connection to a neighbor router, or other significant events.

**Note** Inform requests (inform operations) are supported in Cisco IOS XR software.

Traps are less reliable than informs because the receiver does not send any acknowledgment when it receives a trap. The sender cannot determine if the trap was received. An SNMP manager that receives an inform request acknowledges the message with an SNMP response protocol data unit (PDU). If the manager does not receive an inform request, it does not send a response. If the sender never receives a response, the inform request can be sent again. Thus, informs are more likely to reach their intended destination.

However, traps are often preferred because informs consume more resources in the router and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform request must be held in memory until a response is received or the request times out. Also, traps are sent only once, and an inform may be retried several times. The retries increase traffic and contribute to a higher overhead on the network. Thus, traps and inform requests provide a trade-off between reliability and resources.

***Figure 2: Trap Received by the SNMP Manager***

In this illustration, the agent router sends a trap to the SNMP manager. Although the manager receives the trap, it does not send any acknowledgment to the agent. The agent has no way of knowing that the trap reached its destination.

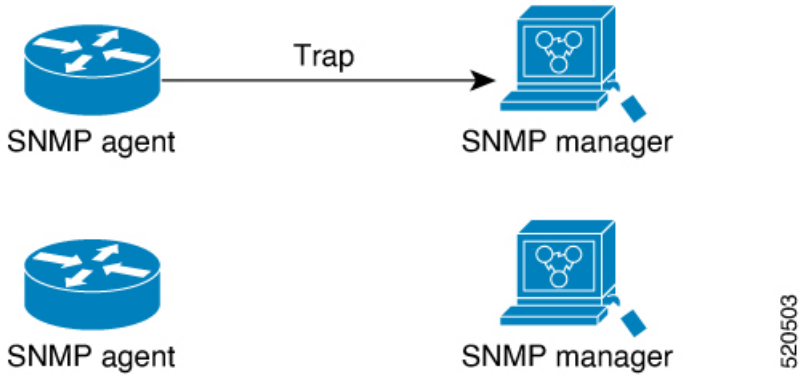
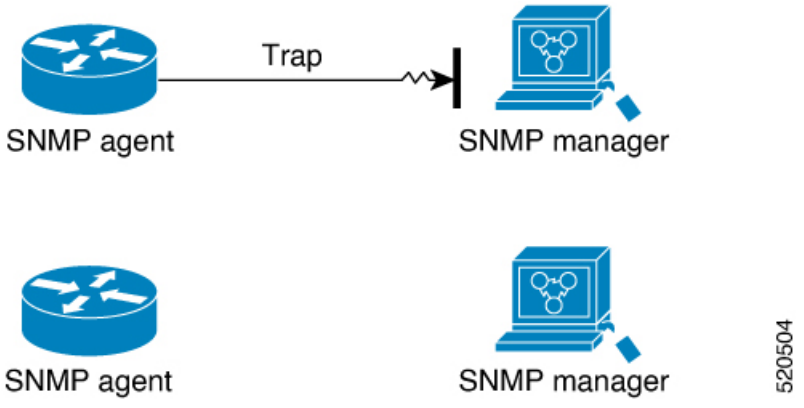



***Figure 3: Trap Not Received by the SNMP Manager***

In this illustration, the agent sends a trap to the manager, but the trap does not reach the manager. Because the agent has no way of knowing that the trap did not reach its destination, the trap is not sent again. The manager never receives the trap.



## Session Types

The supported session types are:

- PPPoE
- IP SUB PKT
- IP SUB DHCP

# How to Implement SNMP on Cisco IOS XR Software

This section describes how to implement SNMP.

The **snmp-server** commands enable SNMP on Management Ethernet interfaces by default. For information on how to enable SNMP server support on other inband interfaces, see the *Implementing Management Plane Protection on Cisco IOS XR Software* module in *System Security Configuration Guide*.

## Configuring SNMPv3

This task explains how to configure SNMPv3 for network management and monitoring.

**Note** No specific command enables SNMPv3; the first **snmp-server** global configuration command (config), that you issue enables SNMPv3. Therefore, the sequence in which you issue the **snmp-server** commands for this task does not matter.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | (Optional) **snmp-server engineid local** *engine-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server engineID local 00:00:00:09:00:00:00:a1:61:6c:20:61` | Specifies the identification number of the local SNMP engine. |
| **Step 3** | **snmp-server view** *view-name oid-tree* {**included** \| **excluded**}<br>**Example:** | Creates or modifies a view record. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# snmp-server view view_name 1.3.6.1.2.1.1.5 included` | |
| **Step 4** | **snmp-server group** *name* {**v1** \| **v2c** \| **v3** {**auth** \| **noauth** \| **priv**}} [**read** *view*] [**write** *view*] [**notify** *view*] [*access-list-name*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server group group_name v3 noauth read view_name1 write view_name2` | Configures a new SNMP group or a table that maps SNMP users to SNMP views. |
| **Step 5** | **snmp-server user** *username groupname* {**v1** \| **v2c** \| **v3** [**auth** {**md5** \| **sha**} {**clear** \| **encrypted**} *auth-password* [**priv des56** {**clear** \| **encrypted**} *priv-password*]]} [*access-list-name*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server user noauthuser group_name v3` | Configures a new user to an SNMP group. |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 7** | (Optional) **show snmp**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp` | Displays information about the status of SNMP. |
| **Step 8** | (Optional) **show snmp engineid**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp engineid` | Displays information about the local SNMP engine. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 9** | (Optional) **show snmp group**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp group` | Displays information about each SNMP group on the network. |
| **Step 10** | (Optional) **show snmp users**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp users` | Displays information about each SNMP username in the SNMP users table. |
| **Step 11** | (Optional) **show snmp view**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp view` | Displays information about the configured views, including the associated MIB view family name, storage type, and status. |

# Configuring SNMPv3: Examples

### Setting an Engine ID

This example shows how to set the identification of the local SNMP engine:

```
config
snmp-server engineID local 00:00:00:09:00:00:00:a1:61:6c:20:61
```

**Note** After the engine ID has been configured, the SNMP agent restarts.

### Verifying the Identification of the Local SNMP Engines

This example shows how to verify the identification of the local SNMP engine:

```
show snmp engineid

SNMP engineID 00000009000000a1ffffffff
```

### Creating a View

There are two ways to create a view:

- You can include the object identifier (OID) of an ASN.1 subtree of a MIB family from a view by using the **included** keyword of the **snmp-server view** command.
- You can exclude the OID subtree of the ASN.1 subtree of a MIB family from a view by using the **excluded** keyword of the **snmp-server view** command.

This example shows how to create a view that includes the sysName (1.3.6.1.2.1.1.5) object:

```
config
  snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1.5 included
```

This example shows how to create a view that includes all the OIDs of a system group:

```
config
  snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1 included
```

This example shows how to create a view that includes all the OIDs under the system group except the sysName object (1.3.6.1.2.1.1.5), which has been excluded:

```
config
  snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1 included
  snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1.5 excluded
```

### Verifying Configured Views

This example shows how to display information about the configured views:

```
RP/0/RP0/CPU0:router# show snmp view

  v1default 1.3.6.1 - included nonVolatile active
  SNMP_VIEW1 1.3.6.1.2.1.1 - included nonVolatile active
  SNMP_VIEW1 1.3.6.1.2.1.1.5 - excluded nonVolatile active
```

### Creating Groups

If you do not explicitly specify a notify, read, or write view, the Cisco IOS XR software uses the v1 default (1.3.6.1). This example shows how to create a group that utilizes the default view:

```
RP/0/RP0/CPU0:router# snmp-server group group-name v3 auth
```

The following configuration example shows how to create a group that has read access to all the OIDs in the system except the sysUpTime object (1.3.6.1.2.1.1.3), which has been excluded from the view applied to the group, but write access only to the sysName object (1.3.6.1.2.1.1.5):

```
!
  snmp-server view view_name1 1.3.6.1.2.1.1 included
  snmp-server view view_name1 1.3.6.1.2.1.1.3 excluded
  snmp-server view view_name2 1.3.6.1.2.1.1.5 included
  snmp-server group group_name1 v3 auth read view_name1 write view_name2
  !
```

### Verifying Groups

This example shows how to verify the attributes of configured groups:

```
RP/0/RP0/CPU0:router# show snmp group

  groupname: group_name1                  security model:usm
  readview : view_name1                   writeview: view_name2
  notifyview: v1default
  row status: nonVolatile
```

### Creating and Verifying Users

Given the following SNMPv3 view and SNMPv3 group configuration:

```
!
  snmp-server view view_name 1.3.6.1.2.1.1 included
  snmp-server group group_name v3 noauth read view_name write view-name
  !
```

This example shows how to create a noAuthNoPriv user with read and write view access to a system group:

```
config
  snmp-server user noauthuser group_name v3
```

**Note** The user must belong to a noauth group before a noAuthNoPriv user can be created.

This example shows how to verify the attributes that apply to the SNMP user:

```
RP/0/RP0/CPU0:router# show snmp user

  User name: noauthuser
  Engine ID: localSnmpID
  storage-type: nonvolatile active
```

Given the following SNMPv3 view and SNMPv3 group configuration:

```
!
  snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1 included
  snmp-server group SNMP_GROUP1 v3 auth notify SNMP_VIEW1 read SNMP_VIEW1 write SNMP_VIEW1
 !
```

This example shows how to create a user with authentication (including encryption), read, and write view access to a system group:

```
config
  snmp-server user userv3authpriv SNMP_GROUP1 v3 auth md5 password123 priv aes 128 password123
```

Given the following SNMPv3 view and SNMPv3 group configuration:

```
!
  snmp-server view view_name 1.3.6.1.2.1.1 included
  snmp group group_name v3 priv read view_name write view_name
  !
```

This example shows how to create authNoPriv user with read and write view access to a system group:

```
RP/0/RP0/CPU0:router# snmp-server user authuser group_name v3 auth md5 clear auth_passwd
```

**Note** Because the group is configured at a security level of Auth, the user must be configured as "auth" at a minimum to access this group ("priv" users could also access this group). The authNoPriv user configured in this group, authuser, must supply an authentication password to access the view. In the example, auth_passwd is set as the authentication password string. Note that **clear** keyword is specified before the auth_passwd password string. The **clear** keyword indicates that the password string being supplied is unencrypted.

This example shows how to verify the attributes that apply to SNMP user:

```
RP/0/RP0/CPU0:router# show snmp user

  User name: authuser
  Engine ID: localSnmpID
  storage-type: nonvolatile active
```

Given the following SNMPv3 view and SNMPv3 group configuration:

```
!
  snmp view view_name 1.3.6.1.2.1.1 included
  snmp group group_name v3 priv read view_name write view_name
  !
```

This example shows how to create an authPriv user with read and write view access to a system group:

```
config
  snmp-server user privuser group_name v3 auth md5 clear auth_passwd priv des56 clear
priv_passwd
```

**Note** Because the group has a security level of Priv, the user must be configured as a "priv" user to access this group. In this example, the user, privuser, must supply both an authentication password and privacy password to access the OIDs in the view.

This example shows how to verify the attributes that apply to the SNMP user:

```
RP/0/RP0/CPU0:router# show snmp user

  User name: privuser
  Engine ID: localSnmpID
  storage-type: nonvolatile active
```

# Configuring SNMP Trap Notifications

This task explains how to configure the router to send SNMP trap notifications.

**Note** You can omit #unique_78 if you have already completed the steps documented under the #unique_78 task.

## Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **snmp-server group** *name* {**v1 v2 v3** {**auth** \| **noauth** \| **priv**}} [**read** *view*] **write** *view*] [**notify** *view*] [*access-list-name*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server group group_name v3 noauth read view_name1 writer view_name2` | Configures a new SNMP group or a table that maps SNMP users to SNMP views. |
| **Step 3** | **snmp-server user** *username groupname* {**v1 v2c v3** {**auth** \| **md5** \| **sha**} {**clear** \| **encrypted**} *auth-password*] [**priv des56** {**clear** \| *access-list-name*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server group group_name v3 noauth read view_name1 writer view_name2` | Configures a new SNMP group or a table that maps SNMP users to SNMP views. |
| **Step 4** | [ **snmp-server host** *address* [**traps**] [**version** {**1** \| **2c** \| **3** [**auth** \| **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server host 12.26.25.61 traps version 3 noauth userV3noauth` | Specifies SNMP trap notifications, the version of SNMP to use, the security level of the notifications, and the recipient (host) of the notifications. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 5** | **snmp-server traps** [*notification-type*]<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server traps bgp` | Enables the sending of trap notifications and specifies the type of trap notifications to be sent.<br>• If a trap is not specified with the *notification-type* argument, all supported trap notifications are enabled on the router. To display which trap notifications are available on your router, enter the **snmp-server traps ?** command. |
| **Step 6** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |
| **Step 7** | (Optional) **show snmp host**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp host` | Displays information about the configured SNMP notification recipient (host), port number, and security model. |

# Configuring Trap Notifications: Example

The following example configures an SNMP agent to send out different types of traps. The configuration includes a v2c user, a noAuthNoPriv user, anauthNoPriv user, and an AuthPriv user.

**Note** The default User Datagram Protocol (UDP) port is 161. If you do not a specify a UDP port with the **udp-port** keyword and *port* argument, then the configured SNMP trap notifications are sent to port 161.

```
!
  snmp-server host 10.50.32.170 version 2c public udp-port 2345
  snmp-server host 10.50.32.170 version 3 auth userV3auth udp-port 2345
  snmp-server host 10.50.32.170 version 3 priv userV3priv udp-port 2345
  snmp-server host 10.50.32.170 version 3 noauth userV3noauth udp-port 2345
  snmp-server user userv2c groupv2c v2c
  snmp-server user userV3auth groupV3auth v3 auth md5 encrypted 140F0A13
 snmp-server user userV3priv groupV3priv v3 auth md5 encrypted 021E1C43 priv des56 encrypted
```

```
1110001C
 snmp-server user userV3noauth groupV3noauth v3 LROwner
 snmp-server view view_name 1.3 included
 snmp-server community public RW
 snmp-server group groupv2c v2c read view_name
 snmp-server group groupV3auth v3 auth read view_name
 snmp-server group groupV3priv v3 priv read view_name
 snmp-server group groupV3noauth v3 noauth read view_name
 !
```

In the following example, the output of the **show snmp host** commaand shows how to verify the configuration SNMP trap notification recipients host, the recipients of SNMP trap notifications. The output displays the following information:

- IP address of the configured notification host
- UDP port where SNMP notification messages are sent
- Type of trap configured
- Security level of the configured user
- Security model configured

```
Notification host: 10.50.32.170 udp-port: 2345 type: trap
user: userV3auth security model: v3 auth

Notification host: 10.50.32.170 udp-port: 2345 type: trap
user: userV3noauth security model: v3 noauth

Notification host: 10.50.32.170 udp-port: 2345 type: trap
user: userV3priv security model: v3 priv

Notification host: 10.50.32.170 udp-port: 2345 type: trap
user: userv2c security model: v2c
```

# Setting the Contact, Location, and Serial Number of the SNMP Agent

This task explains how to set the system contact string, system location string, and system serial number of the SNMP agent.

**Note** The sequence in which you issue the **snmp-server** commands for this task does not matter.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | (Optional) **snmp-server contact** *system-contact-string*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server contact`<br>`Dial System Operator at beeper # 27345` | Sets the system contact string. |
| **Step 3** | (Optional) **snmp-server location** *system-location*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server location`<br>`Building 3/Room 214` | Sets the system location string. |
| **Step 4** | (Optional) **snmp-server chassis-id** *serial-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server chassis-id 1234456` | Sets the system serial number. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Defining the Maximum SNMP Agent Packet Size

This task shows how to configure the largest SNMP packet size permitted when the SNMP server is receiving a request or generating a reply.

**Note** The sequence in which you issue the **snmp-server** commands for this task does not matter.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | (Optional) **snmp-server packetsize** *byte-count*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server packetsize 1024` | Sets the maximum packet size. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Changing Notification Operation Values

After SNMP notifications have been enabled, you can specify a value other than the default for the source interface, message queue length, or retransmission interval.

This task explains how to specify a source interface for trap notifications, the message queue length for each host, and the retransmission interval.

**Note** The sequence in which you issue the **snmp-server** commands for this task does not matter.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:** | Enters mode. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router# configure` | |
| **Step 2** | (Optional) **snmp-server trap-source** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server trap-source POS 0/0/1/0` | Specifies a source interface for trap notifications. |
| **Step 3** | (Optional) **snmp-server queue-length** *length*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server queue-length 20` | Establishes the message queue length for each notification. |
| **Step 4** | (Optional) **snmp-server trap-timeout** *seconds*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server trap-timeout 20` | Defines how often to resend notifications on the retransmission queue. |
| **Step 5** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Setting IP Precedence and DSCP Values

This task describes how to configure IP Precedence or IP DSCP for SNMP traffic.

### Before you begin

SNMP must be configured.

**Procedure**

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | Use one of the following commands:<br>• **snmp-server ipv4 precedence** *value*<br>• **snmp-server ipv4 dscp** *value*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server dscp 24` | Configures an IP precedence or IP DSCP value for SNMP traffic. |
| **Step 3** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration session, without committing the configuration changes. |

# Setting an IP Precedence Value for SNMP Traffic: Example

The following example shows how to set the SNMP IP Precedence value to 7:

```
configure
  snmp-server ipv4 precedence 7
  exit

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: y
```

# Setting an IP DSCP Value for SNMP Traffic: Example

The following example shows how to set the IP DSCP value of SNMP traffic to 45:

```
configure
  snmp-server ipv4 dscp 45
  exit

  Uncommitted changes found, commit them before exiting(yes/no/cancel)? [cancel]: y
```

# Displaying SNMP Context Mapping

The SNMP agent serves queries based on SNMP contexts created by the client features. There is a context mapping table. Each entry in the context mapping table includes a context name, the name of the feature that created the context, and the name of the specific instance of the feature.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **show snmp context-mapping**<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp context-mapping` | Displays the SNMP context mapping table. |

# Monitoring Packet Loss

It is possible to monitor packet loss by configuring the generation of SNMP traps when packet loss exceeds a specified threshold. The configuration described in this task enables the creation of entries in the MIB tables of the EVENT-MIB. This can then be monitored for packet loss using SNMP GET operations.

### Before you begin

**Note** Entries created in the EVENT-MIB MIB tables using the configuration described in this task cannot be altered using an SNMP SET.

Entries to the EVENT-MIB MIB tables created using an SNMP SET cannot be altered using the configuration described in this task.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **snmp-server mibs eventmib packet-loss** *type interface-path-id* **falling** *lower-threshold* **interval** *sampling-interval* **rising** *upper-threshold*<br>**Example:** | Generates SNMP EVENT-MIB traps for the interface when the packet loss exceeds the specified thresholds. Up to 100 interfaces can be monitored. |

| | Command or Action | Purpose |
|---|---|---|
| | `RP/0/RP0/CPU0:router(config)# snmp-server mibs eventmib packet-loss TenGigE0/0/0/0 falling 1 interval 5 rising 2` | **falling** *lower-threshold* —Specifies the lower threshold. When packet loss between two intervals falls below this threshold and an mteTriggerRising trap was generated previously, a SNMP mteTriggerFalling trap is generated. This trap is not generated until the packet loss exceeds the upper threshold and then falls back below the lower threshold.<br>**interval** *sampling-interval* —Specifies how often packet loss statistics are polled. This is a value between 5 and 1440 minutes, in multiples of 5.<br>**rising** *upper-threshold* —Specifies the upper threshold. When packet loss between two intervals increases above this threshold, an SNMP mteTriggreRising trap is generated. This trap is not generated until the packet loss drops below the lower threshold and then rises above the upper threshold. |

# Configuring MIB Data to be Persistent

Many SNMP MIB definitions define arbitrary 32-bit indices for their object tables. MIB implementations often do a mapping from the MIB indices to some internal data structure that is keyed by some other set of data. In these MIB tables the data contained in the table are often other identifiers of the element being modelled. For example, in the ENTITY-MIB, entries in the entPhysicalTable are indexed by the 31-bit value, entPhysicalIndex, but the entities could also be identified by the entPhysicalName or a combination of the other objects in the table.

Because of the size of some MIB tables, significant processing is required to discover all the mappings from the 32-bit MIB indices to the other data which the network management station identifies the entry. For this reason, it may be necessary for some MIB indices to be persistent across process restarts, switchovers, or device reloads. The ENTITY-MIB entPhysicalTable and CISCO-CLASS-BASED-QOS-MIB are two such MIBs that often require index values to be persistent.

Also, because of query response times and CPU utilization during CISCO-CLASS-BASED-QOS-MIB statistics queries, it is desirable to cache service policy statistics.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | (Optional) **snmp-server mibs cbqosmib persist**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server mibs cbqosmib persist` | Enables persistent storage of the CISCO-CLASS-BASED-QOS-MIB data. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 2** | **snmp-server ifindex persist**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server ifindex persist` | Enables ifIndex persistence globally on all Simple Network Management Protocol (SNMP) interfaces. |

# Configuring LinkUp and LinkDown Traps for a Subset of Interfaces

By specifying a regular expression to represent the interfaces for which you are interested in setting traps, you can enable or disable linkUp and linkDown traps for a large number of interfaces simultaneously.

### Before you begin

SNMP must be configured.

### Procedure

| | Command or Action | Purpose |
|---|---|---|
| **Step 1** | **configure**<br>**Example:**<br>`RP/0/RP0/CPU0:router# configure` | Enters mode. |
| **Step 2** | **snmp-server interface subset** *subset-number* **regular-expression** *expression*<br>**Example:**<br>`RP/0/RP0/CPU0:router(config)# snmp-server interface subset 10 regular-expression "^Gig[a-zA-Z]+[0-9/]+\."`<br>`RP/0/RP0/CPU0:router(config-snmp-if-subset)#` | Enters snmp-server interface mode for the interfaces identified by the regular expression.<br>The subset-number argument identifies the set of interfaces, and also assigns a priority to the subset in the event that an interface is included in more than one subset. Lower numbers have higher priority and their configuration takes precedent over interface subsets with higher numbers.<br>The *expression* argument must be entered surrounded by double quotes.<br>Refer to the *Understanding Regular Expressions, Special Characters, and Patterns* module in for more information regarding regular expressions. |
| **Step 3** | **notification linkupdown disable**<br>**Example:**<br>`RP/0/RP0/CPU0:router(config-snmp-if-subset)# notification linkupdown disable` | Disables linkUp and linkDown traps for all interfaces being configured. To enable previously disabled interfaces, use the **no** form of this command. |

| | Command or Action | Purpose |
|---|---|---|
| **Step 4** | Use the **commit** or **end** command. | **commit** —Saves the configuration changes, and remains within the configuration session.<br>**end** —Prompts user to take one of these actions:<br>• **Yes** — Saves configuration changes and exits the configuration session.<br>• **No** —Exits the configuration session without committing the configuration changes.<br>• **Cancel** —Remains in the configuration mode, without committing the configuration changes. |
| **Step 5** | (Optional) **show snmp interface notification subset** *subset-number*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp interface notification subset 10` | Displays the linkUp and linkDown notification status for all interfaces identified by the subset priority. |
| **Step 6** | (Optional) **show snmp interface notification regular-expression** *expression*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp interface notification regular-expression "^Gig[a-zA-Z]+[0-9/]+\."` | Displays the linkUp and linkDown notification status for all interfaces identified by the regular expression. |
| **Step 7** | (Optional) **show snmp interface notification** *type interface-path-id*<br>**Example:**<br>`RP/0/RP0/CPU0:router# show snmp interface notification tengige` | Displays the linkUp and linkDown notification status for the specified interface. |