Simple Network Management Protocol

The chapter provides an overview of SNMP, detailing its architecture, versions, and features, including configuration steps for SNMPv3, creating views and groups, and setting up traps and notifications. It also discusses SNMP security models, session types, and QoS mechanisms, emphasizing its role in network management and monitoring.

SNMP

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.

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.

SNMP Agent

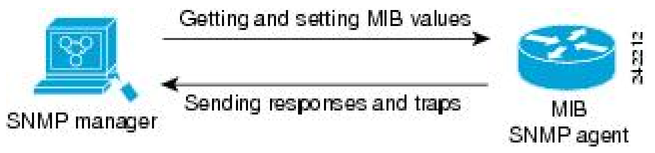
The SNMP agent is the software component within the managed device that maintains the data for the device and reports these data, as needed, to managing systems. The agent and MIB reside on the router. To enable the SNMP agent, you must define the relationship between the manager and the agent.

Management Information Base

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.



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.

These OIDs added to the **ipIfStatsTable**:

- 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.

To determine the MIBs supported, refer to Cisco IOS XR MIBs tool.

Guidelines and Restrictions for SNMP

These are the guidelines and restrictions for use of SNMP:

- SNMP outputs are only 32-bits wide, limiting the display to information less than 2^{32} (4.29 Gigabits), which can cause issues like concatenated results when displaying speed information for interfaces exceeding this limit, such as 10 Gigabit interfaces.
- The recommended maximum number of object identifiers (OIDs) in a single SNMP request is 75. Exceeding this limit can result in SNMP requests being dropped or SNMP polling timeouts.

SNMP Versions

These are the supported 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 Table 1 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.

Features Supported on SNMPv1, SNMPv2c, and SNMPv3

These are the operations supported on SNMPv1, SNMPv2c, and SNMPv3:

- 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.

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

Table 1: SNMPv1, SNMPv2c, and SNMPv3 Feature Support

Feature	SNMPv1	SNMPv2c	SNMPv3
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, SNMPv2, SNMPv3

The security level determines if an SNMP message needs to be protected from disclosure and if the message needs to be authenticated. There are three security levels within a security model:

- 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

SNMP Security 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 (Hash-Based Messsage Authentication Code)-MD5 (Message Digest 5) algorithm or the HMAC-SHA.
v3	authPriv	HMAC-MD5 or HMAC-SHA	DES	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides DES (Data Encryption Standard) 56-bit encryption in addition to authentication based on the CBC (Cipher Block Chaining) 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 (Triple Data Encryption Standard) 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 (Advanced Encryption Standard) level of encryption.

Use of 3DES and AES encryption standards requires that the security package (k9sec) be installed. For information on installing software packages, see Chapter *Upgrading and Managing Cisco IOS XR Software* in *System Setup and Software Installation Guide for ASR 9000 Series Routers*.

SNMPv3 Security Enhancements

SNMPv3 provides secure access to devices by providing authentication, encryption and access control. The added security benefits secure SNMP against these 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.

Comparative Response Times for SNMP Security Models and Levels

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 these 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. These are the three types of access rights:

- 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.

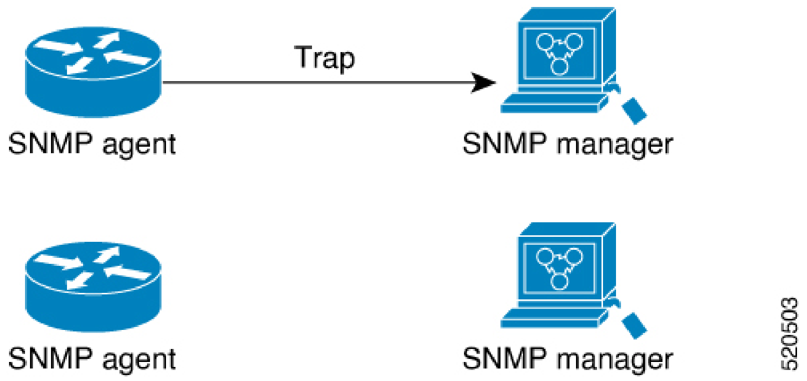
SNMP Notifications: Traps versus Informs

A key feature of SNMP (Simple Network Management Protocol) is the ability to generate notifications from an SNMP agent. SNMP notifications are messages sent by the agent to alert the SNMP manager about specific conditions or events on the network. These notifications do not require requests to be sent from the SNMP manager.

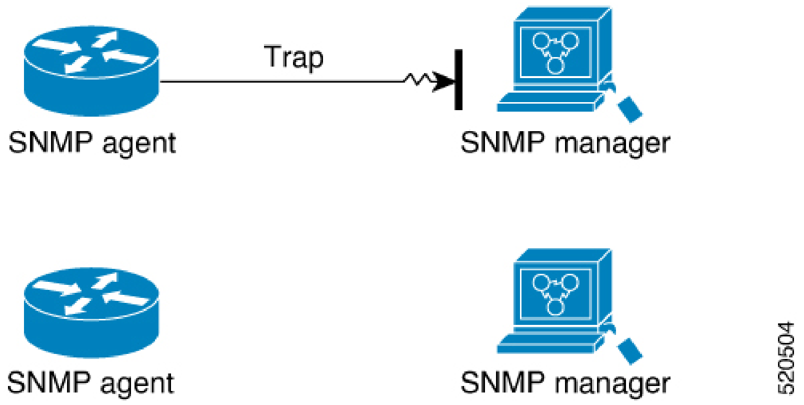
Traps

On Cisco IOS XR software, unsolicited (asynchronous) notifications can be generated only as traps. Traps are a type of SNMP notification that alert the SNMP manager to conditions such as improper user authentication, restarts, the closing of a connection, loss of connection to a neighbor router, or other significant events. Traps are less reliable than informs because the receiver does not send any acknowledgment when it receives a trap, leaving the sender unable to determine if the trap was received. Despite this, traps are often preferred because they consume fewer resources in the router and the network. Once a trap is sent, it is immediately discarded, minimizing memory usage and network traffic. This makes traps suitable for less critical notifications where resource efficiency is more important than guaranteed delivery.

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.



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.



Informs

Informs are another type of SNMP notification that require acknowledgment from the SNMP manager, making them more reliable but also more resource-intensive. 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, prompting the sender to resend the inform. This acknowledgment mechanism ensures that informs are more likely to reach their intended destination. However, informs consume more resources because they must be held in memory until a response is received or the request times out. Additionally, informs may be retried several times, increasing network traffic and contributing to higher overhead. Inform requests (inform operations) are supported from Cisco IOS XR Software Release 4.1 onwards. Thus, informs provide a trade-off between reliability and resource consumption, making them suitable for critical notifications where delivery assurance is essential.

Session Types

These are the supported session types:

- Point-to-Point Protocol over Ethernet (PPPoE) - PPPoE is a network protocol that encapsulates PPP frames inside Ethernet frames. It is commonly used for establishing internet connections over DSL lines. In SNMP, PPPoE sessions can be monitored to check the status, performance, and configuration of connections established using this protocol.
- IP Subscriber Packet (IP SUB PKT) - This typically refers to the handling of IP packets for subscribers in a network, often related to broadband or other IP-based services. Monitoring IP SUB PKT sessions via SNMP can involve tracking packet flow, session status, and subscriber-specific data to ensure proper service delivery and network performance.
- IP Subscriber DHCP (IP SUB DHCP) - This relates to IP address assignment using the DHCP (Dynamic Host Configuration Protocol) for network subscribers. DHCP automatically assigns IP addresses and other network configuration details to devices. SNMP can be used to monitor DHCP sessions, including lease status, address allocation, and overall DHCP server performance.

Configure SNMPv3

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 doesn't matter.

Follow these steps to configure SNMPv3.

Procedure

Step 1 Create or modify a view record.

Example:

```
Router# configure
Router(config)# snmp-server view view_name 1.3.6.1.2.1.1.5 included
```

Step 2 Configure a new SNMP group or a table that maps SNMP users to SNMP views.

Example:

```
Router(config)# snmp-server group group_name v3 noauth read view_name1 write view_name2
```

Step 3 Configure a new user to an SNMP group.

Example:

```
Router(config)# snmp-server user noauthuser group_name v3
Router# commit
```

Note

- Only one remote host can be assigned to the same username for SNMP version 3. If you configure the same username with different remote hosts, only the last username and remote host combination is accepted and is seen in the **show running configuration**. In the case of multiple SNMP managers, multiple unique usernames are required.
- When you execute an SNMP bulk request using the **snmpbulkget** command to an unavailable MIB, it provides a next available MIB.

Set SNMP Engine ID

Follow these steps to set SNMP engine ID:

Procedure

Step 1 Set an engine ID.

Example:

```
Router# configure terminal
Router(config)# snmp-server engineID local 1234567890abcdef1234567890abcdef

Router(conig)# commit
```

Engine ID is unique hexa-decimal string within your network.

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

Step 2 Verify the identification of the local SNMP engine.

Example:

```
Router# show snmp engineid
SNMP engineID 00000009000000a1ffffffff
```

Create SNMP 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.

Follow these steps to create an SNMP view:

Procedure

Step 1 Create a view that includes the sysName(1.3.6.1.2.1.1.5) object.

Example:

```
Router#configure terminal
Router(config)#snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1.5 included
```

Step 2 Create a view that includes all the OIDs of a system group.

Example:

```
Router# configure terminal
Router(config)# snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1 included
```

Step 3 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.

Example:

```
Router# configure terminal
  Router(config)# snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1 included
  Router(config)# snmp-server view SNMP_VIEW1 1.3.6.1.2.1.1.5 excluded
```

Step 4 Display the configured views.

Example:

```
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
```

Create SNMP Group

Follow these steps to create an SNMP group:

Procedure

Step 1 Define the VRF and associate it with an SNMP context.

Example:

```
Router# configure
Router(config)# snmp-server vrf VRF_A context BGP_Context
```

Step 2 Define the SNMP community and associate it with a VRF.

Example:

```
Router(config)# snmp-server community public RW
```

Step 3 Define the SNMP context.

Example:

```
Router(config)#snmp-server context BGP_Context
```

Step 4 Map the community string to the SNMP context.

Example:

```
Router(config)# snmp-server community-map public context BGP_Context
```

Step 5 Specify the SNMP host that will receive the traps.

Example:

```
Router(config)# snmp-server host 198.51.100.1 traps version 2c public
```

Step 6 Verify the SNMP configuration.

Example:

```
Router(config)# show running configuration
snmp-server vrf V1
 context V1_bgp
!
snmp-server community V1 RW
snmp-server context V1_bgp
snmp-server community-map V1 context V1_bgp router bgp 65000
 nsr
 address-family ipv4 unicast
 !
 address-family vpnv4 unicast
 !
neighbor 192.0.2.254
 remote-as 65001
 address-family ipv4 unicast
 route-policy ALL in
 route-policy ALL out
 !
!
vrf V1
**********
 address-family ipv4 unicast
!
neighbor 192.0.2.255
  remote-as 65003
  address-family ipv4 unicast
  !
 !
 !
!
end
```

Configure SNMP Trap Notifications

Follow these steps to configure SNMP trap notifications.

Procedure

Step 1 Configure a new SNMP group or a table that maps SNMP users to SNMP views.

Example:

```
Router# configure
Router(config)# snmp-server group group_name v3 noauth read view_name1 write view_name2
```

Step 2 Configure a new user to an SNMP group.

Example:

```
Router(config)# snmp-server user noauthuser group_name v3
```

Note

- Only one remote host can be assigned to the same username for SNMP version 3. If you configure the same username with different remote hosts, only the last username and remote host combination will be accepted and will be seen in the **show running configuration**. In the case of multiple SNMP managers, multiple unique usernames are required.
- When you execute an SNMP bulk request using the **snmpbulkget** command to an unavailable MIB, it provides a next available MIB.

Step 3 Specify SNMP trap notifications, the version of SNMP to use, the security level of the notifications, and the recipient (host) of the notifications.

Example:

```
Router(config)# snmp-server host 12.26.25.61 traps version 3 noauth userV3noauth
```

Step 4 Enable the sending of trap notifications and specify the type of trap notifications to be sent.

Example:

```
Router(config)# snmp-server traps bgp
Router# commit
```

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 5 Display information about the configured SNMP notification recipient (host), port number, and security model.

Example:

```
Router# show snmp host
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
```

The output gives information about -

- 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, and
- security model configured.

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.

Set the Contact, Location, and Serial Number of the SNMP Agent

Follow these steps to set the contact, location, and serial number of the SNMP Agent.

Procedure

Step 1 (Optional) Set the system contact string.

Example:

```
Router# configure
Router(config)# snmp-server contact Dial System Operator at beeper # 27345
```

Step 2 (Optional) Set the system location string.

Example:

```
Router(config)# snmp-server location Building 3/Room 214
```

Step 3 (Optional) Set the serial number.

Example:

```
Router(config)# snmp-server chassis-id 1234456
Router(config)# commit
```

Set the Maximum SNMP Agent Packet Size

Perform this task to set the maximum SNMP agent packet size.

Procedure

Set the maximum packet size.

Example:

```
Router# configure
Router(config)# snmp-server packetsize 1024
Router(config)# commit
```

Change 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.

Follow these steps to specify a source interface for trap notifications, the message queue length for each host, and the retransmission interval.

Procedure

Step 1 Specify the source interface for trap notifications.

Example:

```
Router# configure
Router(config)# snmp-server trap-source POS 0/0/1/0
```

Step 2 Establish the message queue length for each notication.

Example:

```
Router(config)# snmp-server queue-length 20
```

Defines how often to resend notifications on the queue.

Step 3 Define how often to resend notifications on the retransmission queue.

Example:

```
Router(config)# snmp-server trap-timeout 20
Router(config)# commit
```

Set IP Precedence or DSCP Value

Follow these steps to set IP Precedence or IP DSCP for SNMP traffic.

Before you begin

SNMP must be configured.

Procedure

Set an IP precedence or IP DSCP for SNMP traffic.

Example:

```
Router# configure
Router(config)# snmp-server dscp 24
Router(config)# commit
```

```
Router# configure
Router(config)# snmp-server ipv4 precedence 7
Router(config)# commit
```

Configure 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.

Follow these steps to configure MIB data to be peristent.

Procedure

Step 1 (Optional) Enable the persistent storage of ENTITY-MIB data.

Example:

```
Router(config)# snmp-server entityindex persist
```

Step 2 (Optional) Enable persistent storage of the CISCO-CLASS-BASED-QOS-MIB data.

Example:

```
Router(config)# snmp-server mibs cbqosmib persist
```

Step 3 (Optional) Enable QoS MIB caching with a specified cache refresh time.

Example:

```
Router(config)# snmp-server mibs cbqosmib cache refresh time 45
```

Step 4 (Optional) Enable QoS MIB caching with a limited number of service policies to cache.

Example:

```
Router(config)# snmp-server mibs cbqosmib cache service-policy count 50
```

Step 5 Enable ifIndex persistence globally on all Simple Network Management Protocol (SNMP) interfaces.

Example:

```
Router(config)# snmp-server ifindex persist
```

Configure LinkUp and LinkDown Traps for a Subset of Interfaces

Follow these steps to specify 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

Step 1 Enter snmp-server interface mode for the interfaces identified by the regular expression.

Example:

```
Router# configure
Router(config)# snmp-server interface subset 10 regular-expression "^Gig[a-zA-Z]+[0-9/]+\."
```

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.

The expression argument must be entered surrounded by double quotes.

Step 2 Disable linkUp and linkDown traps for all interfaces being configured using the **notification linkupdown disable** command. To enable previously disabled interfaces, use the **no** form of this command.

Example:

```
Router(config-snmp-if-subset)# notification linkupdown disable
Router(config-snmp-if-subset)# commit
```

Step 3 (Optional) Display the linkUp and linkDown notification status for all interfaces identified by the subset priority.

Example:

```
Router# show snmp interface notification subset 10
```

Step 4 (Optional) Display the linkUp and linkDown notification status for all interfaces identified by the regular expression.

Example:

```
Router# show snmp interface notification regular-expression "^Gig[a-zA-Z]+[0-9/]+\."
```

Step 5 (Optional) Display the linkUp and linkDown notification status for the specified interface.

Example:

```
Router# show snmp interface notification tengige 0/4/0/3.10
```

Configure VRF Aware SNMP Context for Polling BGP Data

VRF awareness is usually done using existing, non-VRF aware MIB definitions. This means that MIB definition doesn't mention anything about VRFs. However they could be used within VRF context. The VRF-awareness is done using SNMP contexts, where a SNMP context maps to a specific VRF.

Follow these steps to configure VRF aware SNMP context for polling BGP data.

Before you begin

- Ensure that MIB implementation is VRF-aware.
- Ensure that the implementation of all get requests support VRF context.

Procedure

Step 1 Define the VRF and associate it with an SNMP context using the **snmp-server vrf context** command.

Example:

```
Router# configure
Router(config)# snmp-server vrf VRF_A context BGP_Context
```

Step 2 Define the SNMP community string and associate it with the VRF using the **snmp-server community RW** command.

Example:

```
Router(config)# snmp-server community public RW
```

Step 3 Define SNMP context using the **snmp-server context** command.

Example:

```
Router(config)# snmp-server context BGP_Context
```

Step 4 Map the community string to the SNMP context using the **snmp-server community-map context** command.

Example:

```
Router(config)# snmp-server community-map public context BGP_Context
```

Step 5 Specify the SNMP host that will receive traps using the **snmp-server host traps version 2c** command.

Example:

```
Router(config)# snmp-server host 198.51.100.1 traps version 2c public
```

Step 6 Verify the configuration using the **show running configuration** command.

Example:

```
Router(config) #show running configuration
snmp-server vrf V1
 context V1_bgp
!
snmp-server community V1 RW
snmp-server context V1_bgp
snmp-server community-map V1 context V1_bgp router bgp 65000
 nsr
 address-family ipv4 unicast
 !
 address-family vpnv4 unicast
 !
neighbor 192.0.2.254
 remote-as 65001
 address-family ipv4 unicast
 route-policy ALL in
 route-policy ALL out
 !
!
vrf V1
 rd 111:111
 address-family ipv4 unicast
!
neighbor 192.0.2.255
  remote-as 65003
  address-family ipv4 unicast
  !
 !
 !
!
end
```

Configure OSPF processes Using SNMP Context

Follow these steps to configure OSPF processes using SNMP context:

Procedure

Step 1 Configure the SNMP communities that are used for read-write access.

Example:

```
Router# configure
Router(config)# snmp-server community com1 RW
Router(config)# snmp-server community com2 RW
```

Step 2 Configure the SNMP contexts that are associated with the OSPF processes.

Example:

```
Router(config)# snmp-server context ctx1
Router(config)# snmp-server context ctx2
```

Step 3 Map the SNMP communities to the respective contexts.

Example:

```
Router(config)# snmp-server community-map com1 context ctx1
Router(config)# snmp-server community-map com2 context ctx2
```

Step 4 Define the first and second OSPF process and associate it with the respective SNMP context.

Example:

```
Router(config)# router ospf one
Router(config-router)# snmp context ctx1
Router(config-router)# area 0
Router(config-router)# interface GigabitEthernet0/2/0/0
Router(config-if)# exit
Router(config-router)# exit

Router(config)# router ospf two
Router(config-router)# snmp context ctx2
Router(config-router)# area 0
Router(config-router)# interface GigabitEthernet0/2/0/1
Router(config-if)# exit
Router(config-router)# exit
```

Configure OSPF Neighbour in VRF

Follow these steps to configure OSPF neighbour in VRF.

Procedure

Step 1 Create the VRF and configure SNMP server for VRF.

Example:

```
Router# configure
Router(config)# vrf definition VRF_A
Router(config)# snmp-server vrf VRF_A
```

Step 2 Create SNMP context.

Example:

```
Router# snmp-server context ctx1
```

Step 3 Configure SNMP Community.

Example:

```
Router(config)# snmp-server community com1 RW
```

Step 4 Map SNMP community with context and associate SNMP context with VRF.

Example:

```
Router# snmp-server community-map com1 context ctx1 router ospf core
Router(config)# vrf VRF_A
Router(config-vrf)# snmp context ctx1
Router(config-vrf)# exit
Router(config)# end
Router# write memory
```