Configuring Simple Network Management Protocol

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest caveats and feature information, see Bug Search Tool and the release notes for your platform and software release. To find information about the features documented in this module, and to see a list of the releases in which each feature is supported, see the feature information table at the end of this module.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn. An account on Cisco.com is not required.

Prerequisites for SNMP

Supported SNMP Versions

This software release supports the following SNMP versions:

- SNMPv1—The Simple Network Management Protocol, a Full Internet Standard, defined in RFC 1157.
- SNMPv2C replaces the Party-based Administrative and Security Framework of SNMPv2Classic with the community-string-based Administrative Framework of SNMPv2C while retaining the bulk retrieval and improved error handling of SNMPv2Classic. It has these features:

- SNMPv2—Version 2 of the Simple Network Management Protocol, a Draft Internet Standard, defined in RFCs 1902 through 1907.
- SNMPv2C—The community-string-based Administrative Framework for SNMPv2, an Experimental Internet Protocol defined in RFC 1901.

- SNMPv3—Version 3 of the SNMP is an interoperable standards-based protocol defined in RFCs 2273 to 2275. SNMPv3 provides secure access to devices by authenticating and encrypting packets over the network and includes these security features:
  - Message integrity—Ensures that a packet was not tampered with in transit.
  - Authentication—Determines that the message is from a valid source.
  - Encryption—Mixes the contents of a package to prevent it from being read by an unauthorized source.

Note To select encryption, enter the **priv** keyword.

Both SNMPv1 and SNMPv2C use a community-based form of security. The community of managers able to access the agent's MIB is defined by an IP address access control list and password.

SNMPv2C includes a bulk retrieval function and more detailed error message reporting to management stations. The bulk retrieval function retrieves tables and large quantities of information, minimizing the number of round-trips required. The SNMPv2C improved error-handling 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 in SNMPv2C report the error type.

SNMPv3 provides for both security models and security levels. A security model is an authentication strategy set up for a user and the group within which the user resides. A security level is the permitted level of security within a security model. A combination of the security level and the security model determine which security method is used when handling an SNMP packet. Available security models are SNMPv1, SNMPv2C, and SNMPv3.

The following table identifies characteristics and compares different combinations of security models and levels:

Table 1: SNMP Security Models and Levels

Model	Level	Authentication	Encryption	Result
SNMPv1	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv2C	noAuthNoPriv	Community string	No	Uses a community string match for authentication.
SNMPv3	noAuthNoPriv	Username	No	Uses a username match for authentication.

Model	Level	Authentication	Encryption	Result
SNMPv3	authNoPriv	Message Digest 5 (MD5) or Secure Hash Algorithm (SHA)	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms.
SNMPv3	authPriv	MD5 or SHA	Data Encryption Standard (DES) or Advanced Encryption Standard (AES)	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Allows specifying the User-based Security Model (USM) with these encryption algorithms: • DES 56-bit encryption in addition to authentication based on the CBC-DES (DES-56) standard. • 3DES 168-bit encryption • AES 128-bit, 192-bit, or 256-bit encryption

You must configure the SNMP agent to use the SNMP version supported by the management station. Because an agent can communicate with multiple managers, you can configure the software to support communications using SNMPv1, SNMPv2C, or SNMPv3.

Restrictions for SNMP

Version Restrictions

- SNMPv1 does not support informs.

SNMPv3 authentication is not supported in the following scenarios:

- If there is a change in the switch priority followed by stack reload.

- If a device with a lower mac address is added to the stack, the device will be elected as the active switch if all the switches in the stack have the same priority.

To avoid SNMPv3 authentication failure, you should manually configure SNMP engineID on the device before SNMPv3 user configuration. With this, the user can manage and administer the device as the user is tied to the engineID.

Information About SNMP

SNMP Overview

SNMP is an application-layer protocol that provides a message format for communication between managers and agents. The SNMP system consists of an SNMP manager, an SNMP agent, and a management information base (MIB). The SNMP manager can be part of a network management system (NMS) such as Cisco Prime Infrastructure. The agent and MIB reside on the device. To configure SNMP on the device, you define the relationship between the manager and the agent.

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

An agent can send unsolicited traps to the manager. Traps are messages alerting the SNMP manager to a condition on the network. Traps can mean improper user authentication, restarts, link status (up or down), MAC address tracking, closing of a TCP connection, loss of connection to a neighbor, or other significant events.

SNMP Manager Functions

The SNMP manager uses information in the MIB to perform the operations described in the following table:

Table 2: SNMP Operations

Operation	Description
get-request	Retrieves a value from a specific variable.
get-next-request	Retrieves a value from a variable within a table.[1]
get-bulk-request[2]	Retrieves large blocks of data, such as multiple rows in a table, that would otherwise require the transmission of many small blocks of data.
get-response	Replies to a get-request, get-next-request, and set-request sent by an NMS.
set-request	Stores a value in a specific variable.
trap	An unsolicited message sent by an SNMP agent to an SNMP manager when some event has occurred.

[1] With this operation, an SNMP manager does not need to know the exact variable name. A sequential search is performed to find the needed variable from within a table.

[2] The get-bulk command only works with SNMPv2 or later.

Note We recommend that the SNMP Manager exclude the **ciscoFlashFileDate** MIB object from its query, to avoid performance related issues. This is because, though the **ciscoFlashFileDate** object is published in the MIB, it is not supported on the product.

SNMP Agent Functions

The SNMP agent responds to SNMP manager requests as follows:

- Get a MIB variable—The SNMP agent begins this function in response to a request from the NMS. The agent retrieves the value of the requested MIB variable and responds to the NMS with that value.
- Set a MIB variable—The SNMP agent begins this function in response to a message from the NMS. The SNMP agent changes the value of the MIB variable to the value requested by the NMS.

The SNMP agent also sends unsolicited trap messages to notify an NMS that a significant event has occurred on the agent. Examples of trap conditions include, but are not limited to, when a port or module goes up or down, when spanning-tree topology changes occur, and when authentication failures occur.

SNMP Community Strings

SNMP community strings authenticate access to MIB objects and function as embedded passwords. In order for the NMS to access the device, the community string definitions on the NMS must match at least one of the three community string definitions on the device.

A community string can have one of the following attributes:

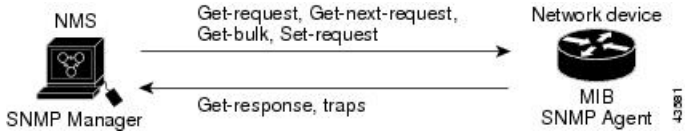
- Read-only (RO)—Gives all objects in the MIB except the community strings read access to authorized management stations, but does not allow write access.
- Read-write (RW)—Gives all objects in the MIB read and write access to authorized management stations, but does not allow access to the community strings.
- When a cluster is created, the command device manages the exchange of messages among member devices and the SNMP application. The Network Assistant software appends the member device number (@esN, where N is the device number) to the first configured RW and RO community strings on the command device and propagates them to the member devices.

SNMP MIB Variables Access

An example of an NMS is the Cisco Prime Infrastructure network management software. Cisco Prime Infrastructure 2.0 software uses the device MIB variables to set device variables and to poll devices on the network for specific information. The results of a poll can be displayed as a graph and analyzed to troubleshoot internetworking problems, increase network performance, verify the configuration of devices, monitor traffic loads, and more.

As shown in the figure, the SNMP agent gathers data from the MIB. The agent can send traps, or notification of certain events, to the SNMP manager, which receives and processes the traps. Traps alert the SNMP manager to a condition on the network such as improper user authentication, restarts, link status (up or down), MAC address tracking, and so forth. The SNMP agent also responds to MIB-related queries sent by the SNMP manager in *get-request*, *get-next-request*, and *set-request* format.

Figure 1: SNMP Network



SNMP Notifications

SNMP allows the device to send notifications to SNMP managers when particular events occur. SNMP notifications can be sent as traps or inform requests. In command syntax, unless there is an option in the command to select either traps or informs, the keyword traps refers to either traps or informs, or both. Use the **snmp-server host** command to specify whether to send SNMP notifications as traps or informs.

Note SNMPv1 does not support informs.

Traps are unreliable because the receiver does not send an acknowledgment when it receives a trap, and the sender cannot determine if the trap was received. When an SNMP manager receives an inform request, it acknowledges the message with an SNMP response protocol data unit (PDU). If the sender does not receive a response, the inform request can be sent again. Because they can be resent, informs are more likely than traps to reach their intended destination.

The characteristics that make informs more reliable than traps also consume more resources in the device and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform request is held in memory until a response is received or the request times out. Traps are sent only once, but an inform might be resent or retried several times. The retries increase traffic and contribute to a higher overhead on the network. Therefore, traps and informs require a trade-off between reliability and resources. If it is important that the SNMP manager receive every notification, use inform requests. If traffic on the network or memory in the device is a concern and notification is not required, use traps.

SNMP ifIndex MIB Object Values

The SNMP agent's IF-MIB module comes up shortly after reboot. As various physical interface drivers are initialized they register with the IF-MIB module, essentially saying "Give me an ifIndex number". The IF-MIB module assigns the next available ifIndex number on a first-come-first-served basis. That is, minor differences in driver initialization order from one reboot to another can result in the same physical interface getting a different ifIndex number than it had before the reboot (unless ifIndex persistency is enabled of course).

Default SNMP Configuration

Feature	Default Setting
SNMP agent	Disabled[3].
SNMP trap receiver	None configured.
SNMP traps	None enabled except the trap for TCP connections (tty).
SNMP version	If no version keyword is present, the default is Version 1.

Feature	Default Setting
SNMPv3 authentication	If no keyword is entered, the default is the **noauth** (noAuthNoPriv) security level.
SNMP notification type	If no type is specified, all notifications are sent.

[3] This is the default when the device starts and the startup configuration does not have any **snmp-server** global configuration commands.

SNMP Configuration Guidelines

If the device starts and the device startup configuration has at least one **snmp-server** global configuration command, the SNMP agent is enabled.

An SNMP *group* is a table that maps SNMP users to SNMP views. An SNMP *user* is a member of an SNMP group. An SNMP *host* is the recipient of an SNMP trap operation. An SNMP *engine ID* is a name for the local or remote SNMP engine.

When configuring SNMP, follow these guidelines:

- When configuring an SNMP group, do not specify a notify view. The **snmp-server host** global configuration command auto-generates a notify view for the user and then adds it to the group associated with that user. Modifying the group's notify view affects all users associated with that group.
- To configure a remote user, specify the IP address or port number for the remote SNMP agent of the device where the user resides.
- Before you configure remote users for a particular agent, configure the SNMP engine ID, using the **snmp-server engineID** global configuration command with the **remote** option. The remote agent's SNMP engine ID and user password are used to compute the authentication and privacy digests. If you do not configure the remote engine ID first, the configuration command fails.
- When configuring SNMP informs, you need to configure the SNMP engine ID for the remote agent in the SNMP database before you can send proxy requests or informs to it.
- If a local user is not associated with a remote host, the device does not send informs for the **auth** (authNoPriv) and the **priv** (authPriv) authentication levels.
- Changing the value of the SNMP engine ID has significant results. A user's password (entered on the command line) is converted to an MD5 or SHA security digest based on the password and the local engine ID. The command-line password is then destroyed, as required by RFC 2274. Because of this deletion, if the value of the engine ID changes, the security digests of SNMPv3 users become invalid, and you need to reconfigure SNMP users by using the **snmp-server user** *username* global configuration command. Similar restrictions require the reconfiguration of community strings when the engine ID changes.

How to Configure SNMP

Configuring Community Strings

You use the SNMP community string to define the relationship between the SNMP manager and the agent. The community string acts like a password to permit access to the agent on the device. Optionally, you can specify one or more of these characteristics associated with the string:

- An access list of IP addresses of the SNMP managers that are permitted to use the community string to gain access to the agent
- A MIB view, which defines the subset of all MIB objects accessible to the given community
- Read and write or read-only permission for the MIB objects accessible to the community

Follow these steps to configure a community string on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [*access-list-number*] **Example:** `Device(config)# snmp-server community comaccess ro 4`	Configures the community string. **Note** The @ symbol is used for delimiting the context information. Avoid using the @ symbol as part of the SNMP community string when configuring this command. • For *string*, specify a string that acts like a password and permits access to the SNMP protocol. You can configure one or more community strings of any length. • (Optional) For **view**, specify the view record accessible to the community. • (Optional) Specify either read-only (**ro**) if you want authorized management stations

	Command or Action	Purpose
		to retrieve MIB objects, or specify read-write (**rw**) if you want authorized management stations to retrieve and modify MIB objects. By default, the community string permits read-only access to all objects. • (Optional) For *access-list-number*, enter an IP standard access list numbered from 1 to 99 and 1300 to 1999.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 4 deny any`	(Optional) If you specified an IP standard access list number in Step 3, then create the list, repeating the command as many times as necessary. • For *access-list-number*, enter the access list number specified in Step 3. • The **deny** keyword denies access if the conditions are matched. The **permit** keyword permits access if the conditions are matched. • For *source*, enter the IP address of the SNMP managers that are permitted to use the community string to gain access to the agent. • (Optional) For *source-wildcard*, enter the wildcard bits in dotted decimal notation to be applied to the source. Place ones in the bit positions that you want to ignore. Recall that the access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:**	(Optional) Saves your entries in the configuration file.

	Command or Action	Purpose
	`Device# copy running-config startup-config`	

What to do next

To disable access for an SNMP community, set the community string for that community to the null string (do not enter a value for the community string).

To remove a specific community string, use the **no snmp-server** community string global configuration command.

You can specify an identification name (engine ID) for the local or remote SNMP server engine on the device. You can configure an SNMP server group that maps SNMP users to SNMP views, and you can add new users to the SNMP group.

Configuring SNMP Groups and Users

You can specify an identification name (engine ID) for the local or remote SNMP server engine on the device. You can configure an SNMP server group that maps SNMP users to SNMP views, and you can add new users to the SNMP group.

Follow these steps to configure SNMP groups and users on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server engineID** {**local** *engineid-string* | **remote** *ip-address* [**udp-port** *port-number*] *engineid-string*} **Example:** `Device(config)# snmp-server engineID local 1234`	Configures a name for either the local or remote copy of SNMP. • The *engineid-string* is a 24-character ID string with the name of the copy of SNMP. You need not specify the entire 24-character engine ID if it has trailing zeros. Specify only the portion of the engine ID up to the point where only zeros remain in the value. The Step Example

	Command or Action	Purpose
		configures an engine ID of 123400000000000000000000. • If you select **remote**, specify the *ip-address* of the device that contains the remote copy of SNMP and the optional User Datagram Protocol (UDP) port on the remote device. The default is 162.
Step 4	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **Example:** `Device(config)# snmp-server group public v2c access lmnop`	Configures a new SNMP group on the remote device. For *group-name*, specify the name of the group. Specify one of the following security models: • **v1** is the least secure of the possible security models. • **v2c** is the second least secure model. It allows transmission of informs and integers twice the normal width. • **v3**, the most secure, requires you to select one of the following authentication levels: **auth**—Enables the Message Digest 5 (MD5) and the Secure Hash Algorithm (SHA) packet authentication. **noauth**—Enables the noAuthNoPriv security level. This is the default if no keyword is specified. **priv**—Enables Data Encryption Standard (DES) packet encryption (also called privacy). (Optional) Enter **read** *readview* with a string (not to exceed 64 characters) that is the name of the view in which you can only view the contents of the agent. (Optional) Enter **write** *writeview* with a string (not to exceed 64 characters) that is the name of the view in which you enter data and configure the contents of the agent. (Optional) Enter **notify** *notifyview* with a string (not to exceed 64 characters) that is the name of the view in which you specify a notify, inform, or trap.

	Command or Action	Purpose
		(Optional) Enter **access** *access-list* with a string (not to exceed 64 characters) that is the name of the access list.
Step 5	**snmp-server user** *username group-name* {**remote** *host* [**udp-port** *port*] } {**v1** [**access** *access-list*] | **v2c** [**access** *access-list*] | **v3** [**encrypted**] [**access** *access-list*] [**auth** {**md5** | **sha**} *auth-password*] } [*priv* {**des** | **3des** | **aes** {**128** | **192** | **256**}} *priv-password*] **Example:** `Device(config)# snmp-server user Pat public v2c`	Adds a new user for an SNMP group. The *username* is the name of the user on the host that connects to the agent. The *group-name* is the name of the group to which the user is associated. Enter **remote** to specify a remote SNMP entity to which the user belongs and the hostname or IP address of that entity with the optional UDP port number. The default is 162. Enter the SNMP version number (**v1**, **v2c**, or **v3**). If you enter **v3**, you have these additional options: • **encrypted** specifies that the password appears in encrypted format. This keyword is available only when the **v3** keyword is specified. • **auth** is an authentication level setting session that can be either the HMAC-MD5-96 (**md5**) or the HMAC-SHA-96 (**sha**) authentication level and requires a password string *auth-password* (not to exceed 64 characters). If you enter **v3** you can also configure a private (**priv**) encryption algorithm and password string *priv-password* using the following keywords (not to exceed 64 characters): • **priv** specifies the User-based Security Model (USM). • **des** specifies the use of the 56-bit DES algorithm. • **3des** specifies the use of the 168-bit DES algorithm. • **aes** specifies the use of the DES algorithm. You must select either 128-bit, 192-bit, or 256-bit encryption.

	Command or Action	Purpose
		(Optional) Enter **access** *access-list* with a string (not to exceed 64 characters) that is the name of the access list.
Step 6	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 7	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 8	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring SNMP Notifications

A trap manager is a management station that receives and processes traps. Traps are system alerts that the device generates when certain events occur. By default, no trap manager is defined, and no traps are sent. Devices running this Cisco IOS release can have an unlimited number of trap managers.

Note Many commands use the word **traps** in the command syntax. Unless there is an option in the command to select either traps or informs, the keyword **traps** refers to traps, informs, or both. Use the **snmp-server host** global configuration command to specify whether to send SNMP notifications as traps or informs.

You can use the **snmp-server enable traps** global configuration command combined with the **snmp-server host** global configuration command for a specific host to receive the notification types listed in the following table. You can enable any or all of these traps and configure a trap manager to receive them.

Note The **snmp-server enable traps** command does not support traps for local-authentication on your device.

Table 3: Device Notification Types

Notification Type Keyword	Description
bridge	Generates STP bridge MIB traps.

Notification Type Keyword	Description
cluster	Generates a trap when the cluster configuration changes.
config	Generates a trap for SNMP configuration changes.
copy-config	Generates a trap for SNMP copy configuration changes.
cpu threshold	Allow CPU-related traps.
entity	Generates a trap for SNMP entity changes.
envmon	Generates environmental monitor traps. You can enable any or all of these environmental traps: fan, shutdown, status, supply, temperature.
flash	Generates SNMP FLASH notifications. In a device stack, you can optionally enable notification for flash insertion or removal, which would cause a trap to be issued whenever a device in the stack is removed or inserted (physical removal, power cycle, or reload).
fru-ctrl	Generates entity field-replaceable unit (FRU) control traps. In the device stack, this trap refers to the insertion or removal of a device in the stack.
hsrp	Generates a trap for Hot Standby Router Protocol (HSRP) changes.
ipmulticast	Generates a trap for IP multicast routing changes.
mac-notification	Generates a trap for MAC address notifications.
ospf	Generates a trap for Open Shortest Path First (OSPF) changes. You can enable any or all of these traps: Cisco specific, errors, link-state advertisement, rate limit, retransmit, and state changes.
pim	Generates a trap for Protocol-Independent Multicast (PIM) changes. You can enable any or all of these traps: invalid PIM messages, neighbor changes, and rendezvous point (RP)-mapping changes.
port-security	Generates SNMP port security traps. You can also set a maximum trap rate per second. The range is from 0 to 1000; the default is 0, which means that there is no rate limit. **Note** When you configure a trap by using the notification type **port-security**, configure the port security trap first, and then configure the port security trap rate: 1. **snmp-server enable traps port-security** 2. **snmp-server enable traps port-security trap-rate** *rate*
snmp	Generates a trap for SNMP-type notifications for authentication, cold start, warm start, link up or link down.
storm-control	Generates a trap for SNMP storm-control. You can also set a maximum trap rate per minute. The range is from 0 to 1000; the default is 0 (no limit is imposed; a trap is sent at every occurrence).
stpx	Generates SNMP STP Extended MIB traps.

Notification Type Keyword	Description
syslog	Generates SNMP syslog traps.
tty	Generates a trap for TCP connections. This trap is enabled by default.
vlan-membership	Generates a trap for SNMP VLAN membership changes.
vlancreate	Generates SNMP VLAN created traps.
vlandelete	Generates SNMP VLAN deleted traps.
vtp	Generates a trap for VLAN Trunking Protocol (VTP) changes.

Follow these steps to configure the device to send traps or informs to a host.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server engineID remote** *ip-address engineid-string* **Example:** `Device(config)# snmp-server engineID remote 192.180.1.27 00000063000100a1c0b4011b`	Specifies the engine ID for the remote host.
Step 4	**snmp-server user** *username group-name* {**remote** *host* [**udp-port** *port*] } {**v1** [**access** *access-list*] | **v2c** [**access** *access-list*] | **v3** [**encrypted**] [**access** *access-list*] [**auth** {**md5** | **sha**} *auth-password*] } **Example:** `Device(config)# snmp-server user Pat public v2c`	Configures an SNMP user to be associated with the remote host created in Step 3. **Note** You cannot configure a remote user for an address without first configuring the engine ID for the remote host. Otherwise, you receive an error message, and the command is not executed.
Step 5	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **Example:**	Configures an SNMP group.

	Command or Action	Purpose
	`Device(config)# snmp-server group public v2c access lmnop`	
Step 6	**snmp-server host** *host-addr* [**informs** | **traps**] [**version** {**1** | **2c** | **3** {**auth** | **noauth** | **priv**}}] *community-string* [*notification-type*] **Example:** `Device(config)# snmp-server host 203.0.113.1 comaccess snmp`	Specifies the recipient of an SNMP trap operation. For *host-addr*, specify the name or Internet address of the host (the targeted recipient). (Optional) Specify **traps** (the default) to send SNMP traps to the host. (Optional) Specify **informs** to send SNMP informs to the host. (Optional) Specify the SNMP **version** (**1**, **2c**, or **3**). SNMPv1 does not support informs. (Optional) For Version 3, select authentication level **auth**, **noauth**, or **priv**. **Note** The **priv** keyword is available only when the cryptographic software image is installed. For *community-string*, when **version 1** or **version 2c** is specified, enter the password-like community string sent with the notification operation. When **version 3** is specified, enter the SNMPv3 username. The @ symbol is used for delimiting the context information. Avoid using the @ symbol as part of the SNMP community string when configuring this command. (Optional) For *notification-type*, use the keywords listed in the table above. If no type is specified, all notifications are sent.
Step 7	**snmp-server enable traps** *notification-types* **Example:** `Device(config)# snmp-server enable traps snmp`	Enables the device to send traps or informs and specifies the type of notifications to be sent. For a list of notification types, see the table above, or enter **snmp-server enable traps ?** To enable multiple types of traps, you must enter a separate **snmp-server enable traps** command for each trap type.

	Command or Action	Purpose
		Note When you configure a trap by using the notification type **port-security**, configure the port security trap first, and then configure the port security trap rate: **a. snmp-server enable traps port-security** **b. snmp-server enable traps port-security trap-rate** *rate*
Step 8	**snmp-server trap-source** *interface-id* **Example:** `Device(config)# snmp-server trap-source gigabitethernet 1/0/1`	(Optional) Specifies the source interface, which provides the IP address for the trap message. This command also sets the source IP address for informs.
Step 9	**snmp-server queue-length** *length* **Example:** `Device(config)# snmp-server queue-length 20`	(Optional) Establishes the message queue length for each trap host. The range is 1 to 5000; the default is 10.
Step 10	**snmp-server trap-timeout** *seconds* **Example:** `Device(config)# snmp-server trap-timeout 60`	(Optional) Defines how often to resend trap messages. The range is 1 to 1000; the default is 30 seconds.
Step 11	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 12	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 13	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

What to do next

The **snmp-server host** command specifies which hosts receive the notifications. The **snmp-server enable traps** command globally enables the method for the specified notification (for traps and informs). To enable a host to receive an inform, you must configure an **snmp-server host informs** command for the host and globally enable informs by using the **snmp-server enable traps** command.

To remove the specified host from receiving traps, use the **no snmp-server host** *host* global configuration command. The **no snmp-server host** command with no keywords disables traps, but not informs, to the host. To disable informs, use the **no snmp-server host informs** global configuration command. To disable a specific trap type, use the **no snmp-server enable traps** *notification-types* global configuration command.

Setting the Agent Contact and Location Information

Follow these steps to set the system contact and location of the SNMP agent so that these descriptions can be accessed through the configuration file.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server contact** *text* **Example:** `Device(config)# snmp-server contact Dial System Operator at beeper 21555`	Sets the system contact string.
Step 4	**snmp-server location** *text* **Example:** `Device(config)# snmp-server location Building 3/Room 222`	Sets the system location string.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:**	Verifies your entries.

	Command or Action	Purpose
	`Device# show running-config`	
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Limiting TFTP Servers Used Through SNMP

Follow these steps to limit the TFTP servers used for saving and loading configuration files through SNMP to the servers specified in an access list.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server tftp-server-list** *access-list-number* **Example:** `Device(config)# snmp-server tftp-server-list 44`	Limits the TFTP servers used for configuration file copies through SNMP to the servers in the access list. For *access-list-number*, enter an IP standard access list numbered from 1 to 99 and 1300 to 1999.
Step 4	**access-list** *access-list-number* {**deny** | **permit**} *source* [*source-wildcard*] **Example:** `Device(config)# access-list 44 permit 10.1.1.2`	Creates a standard access list, repeating the command as many times as necessary. For *access-list-number*, enter the access list number specified in Step 3. The **deny** keyword denies access if the conditions are matched. The **permit** keyword permits access if the conditions are matched. For *source*, enter the IP address of the TFTP servers that can access the device.

	Command or Action	Purpose
		(Optional) For *source-wildcard*, enter the wildcard bits, in dotted decimal notation, to be applied to the source. Place ones in the bit positions that you want to ignore. The access list is always terminated by an implicit deny statement for everything.
Step 5	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.
Step 6	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 7	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Configuring Trap Flags for SNMP

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 2	**trapflags ap** { **interfaceup** | **register**} **Example:** `Device(config)# trapflags ap interfaceup`	Enables sending AP-related traps. Use the **no** form of the command to disable the trap flags. • **interfaceup**– Enables trap when a Cisco AP interface (A or B) comes up. • **register**– Enables trap when a Cisco AP registers with a Cisco device.

	Command or Action	Purpose
Step 3	**trapflags client** {**dot11	excluded**} **Example:** `Device(config)# trapflags client excluded`	Enables sending client-related dot11 traps. Use the **no** form of the command to disable the trap flags. • **dot11**– Enables Dot11 traps for clients. • **excluded**– Enables excluded traps for clients.
Step 4	**trapflags dot11-security** {**ids-sig-attack	wep-decrypt-error**} **Example:** `Device(config)# trapflags dot11-security wep-decrypt-error`	Enables sending 802.11 security-related traps. Use the **no** form of the command to disable the trap flags. • **ids-sig-attack**– Enables IDS signature attack traps. • **wep-decrypt-error**– Enables traps for WEP decrypt error for clients.
Step 5	**trapflags mesh** **Example:** `Device(config)# trapflags mesh`	Enables trap for the mesh. Use the **no** form of the command to disable the trap flags.
Step 6	**trapflags rogueap** **Example:** `Device(config)# trapflags rogueap`	Enables trap for rogue AP detection. Use the **no** form of the command to disable the trap flags.
Step 7	**trapflags rrm-params** {**channels	tx-power**} **Example:** `Device(config)# trapflags rrm-params tx-power`	Enables sending RRM-parameter update-related traps. Use the **no** form of the command to disable the trap flags. • **channels**– Enables trap when RF Manager automatically changes a channel number for the Cisco AP interface. • **tx-power**– Enables the trap when RF Manager automatically changes Tx-Power level for the Cisco AP interface.
Step 8	**trapflags rrm-profile** {**coverage	interference	load	noise**} **Example:** `Device(config)# trapflags rrm-profile interference`	Enables sending RRM-profile-related traps. Use the **no** form of the command to disable the trap flags. • **coverage**– Enables the trap when the coverage profile maintained by RF Manager fails. • **interference**– Enables the trap when the interference profile maintained by RF Manager fails.

	Command or Action	Purpose
		• **load**– Enables trap when the load profile maintained by RF Manager fails. • **noise**– Enables trap when the noise profile maintained by RF Manager fails.
Step 9	**end** **Example:** `Device(config)# end`	Returns to privileged EXEC mode.

Disabling the SNMP Agent

The **no snmp-server** global configuration command disables all running versions (Version 1, Version 2C, and Version 3) of the SNMP agent on the device. You reenable all versions of the SNMP agent by the first **snmp-server** global configuration command that you enter. There is no Cisco IOS command specifically designated for enabling SNMP.

Follow these steps to disable the SNMP agent.

Before you begin

The SNMP Agent must be enabled before it can be disabled. The SNMP agent is enabled by the first **snmp-server** global configuration command entered on the device.

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**no snmp-server** **Example:** `Device(config)# no snmp-server`	Disables the SNMP agent operation.
Step 4	**end** **Example:**	Returns to privileged EXEC mode.

	Command or Action	Purpose
	`Device(config)# end`	
Step 5	**show running-config** **Example:** `Device# show running-config`	Verifies your entries.
Step 6	**copy running-config startup-config** **Example:** `Device# copy running-config startup-config`	(Optional) Saves your entries in the configuration file.

Monitoring SNMP Status

To display SNMP input and output statistics, including the number of illegal community string entries, errors, and requested variables, use the **show snmp** privileged EXEC command. You also can use the other privileged EXEC commands listed in the table to display SNMP information.

Table 4: Commands for Displaying SNMP Information

Command	Purpose
show snmp	Displays SNMP statistics.
show snmp group	Displays information on each SNMP group on the network.
show snmp pending	Displays information on pending SNMP requests.
show snmp sessions	Displays information on the current SNMP sessions.
show snmp user	Displays information on each SNMP user name in the SNMP us **Note** You must use this command to display SNMPv3 co information for **auth** | **noauth** | **priv** mode. This infor displayed in the **show running-config** output.

SNMP Examples

This example shows how to enable all versions of SNMP. The configuration permits any SNMP manager to access all objects with read-only permissions using the community string *public*. This configuration does not cause the device to send any traps.

```
Device(config)# snmp-server community public
```

This example shows how to permit any SNMP manager to access all objects with read-only permission using the community string *public*. The device also sends VTP traps to the hosts 192.180.1.111 and 192.180.1.33 using SNMPv1 and to the host 192.180.1.27 using SNMPv2C. The community string *public* is sent with the traps.

```
Device(config)# snmp-server community public
Device(config)# snmp-server enable traps vtp
Device(config)# snmp-server host 192.180.1.27 version 2c public
Device(config)# snmp-server host 192.180.1.111 version 1 public
Device(config)# snmp-server host 192.180.1.33 public
```

This example shows how to allow read-only access for all objects to members of access list 4 that use the *comaccess* community string. No other SNMP managers have access to any objects. SNMP Authentication Failure traps are sent by SNMPv2C to the host *cisco.com* using the community string *public*.

```
Device(config)# snmp-server community comaccess ro 4
Device(config)# snmp-server enable traps snmp authentication
Device(config)# snmp-server host cisco.com version 2c public
```

This example shows how to send Entity MIB traps to the host *cisco.com*. The community string is restricted. The first line enables the device to send Entity MIB traps in addition to any traps previously enabled. The second line specifies the destination of these traps and overwrites any previous **snmp-server** host commands for the host *cisco.com*.

```
Device(config)# snmp-server enable traps entity
Device(config)# snmp-server host cisco.com restricted entity
```

This example shows how to enable the device to send all traps to the host *myhost.cisco.com* using the community string *public*:

```
Device(config)# snmp-server enable traps
Device(config)# snmp-server host myhost.cisco.com public
```

This example shows how to associate a user with a remote host and to send **auth** (authNoPriv) authentication-level informs when the user enters global configuration mode:

```
Device(config)# snmp-server engineID remote 192.180.1.27 00000063000100a1c0b4011b
Device(config)# snmp-server group authgroup v3 auth
Device(config)# snmp-server user authuser authgroup remote 192.180.1.27 v3 auth md5 mypassword
Device(config)# snmp-server user authuser authgroup v3 auth md5 mypassword
Device(config)# snmp-server host 192.180.1.27 informs version 3 auth authuser config
Device(config)# snmp-server enable traps
Device(config)# snmp-server inform retries 0
```

Feature History and Information for Simple Network Management Protocol

Release	Modification
Cisco IOS XE 3.3SECisco IOS XE 3.3SE	This feature was introduced.