



Configuring SNMP

This chapter describes how to configure the Simple Network Management Protocol (SNMP) on the Catalyst 6500 series switches.

This chapter consists of these sections:

- [SNMP Terminology, page 38-1](#)
- [Understanding How SNMP Works, page 38-4](#)
- [Understanding How SNMPv1 and SNMPv2c Work, page 38-5](#)
- [Understanding How SNMPv3 Works, page 38-7](#)
- [Enabling and Disabling SNMP Processing, page 38-10](#)
- [Configuring SNMPv1 and SNMPv2c on the Switch, page 38-11](#)
- [SNMPv1 and SNMPv2c Enhancements in Software Release 7.5\(1\), page 38-12](#)
- [Configuring SNMPv3 on the Switch, page 38-16](#)



Note

For complete syntax and usage information for the commands used in this chapter, refer to the *Catalyst 6500 Series Switch Command Reference* publication.

SNMP Terminology

[Table 38-1](#) lists the terms that are used in SNMP technology.

Table 38-1 SNMP Terminology

Term	Definition
authentication	The process of ensuring message integrity and protection against message replays, including both data integrity and data origin authentication.
authoritative SNMP engine	One of the SNMP copies involved in network communication is designated the allowed SNMP engine to protect against message replay, delay, and redirection. The security keys used for authenticating and encrypting SNMPv3 packets are generated as a function of the authoritative SNMP engine's ID and user passwords. When an SNMP message expects a response (for example, get exact, get next, set request), the <i>receiver</i> of these messages is authoritative. When an SNMP message does not expect a response, the <i>sender</i> is authoritative.
community string	A text string used to authenticate messages between a management station and an SNMPv1 or SNMPv2c engine.
data integrity	A condition or state of data in which a message packet has not been altered or destroyed in an unauthorized manner.
data origin authentication	The ability to verify the identity of a user on whose behalf the message is supposedly sent. This ability protects users against both message capture and replay by a different SNMP engine and against packets received or sent to a particular user that uses an incorrect password or security level.
encryption	A method of hiding data from an unauthorized user by scrambling the contents of an SNMP packet.
group	A set of users belonging to a particular security model. A group defines the access rights for all the users belonging to it. Access rights define the SNMP objects that can be read, written to, or created. In addition, the group defines the notifications that a user is allowed to receive.
notification host	An SNMP entity to which notifications (traps and informs) are to be sent.
notify view	A view name (not to exceed 64 characters) for each group; the view name defines the list of notifications that can be sent to each user in the group.
privacy	An encrypted state of the contents of an SNMP packet; in this state the contents are prevented from being disclosed on a network. Encryption is performed with an algorithm called CBC-DES (DES-56).
read view	A view name (not to exceed 64 characters) for each group; the view name defines the list of object identifiers (OIDs) that can be read by users belonging to the group.

Table 38-1 SNMP Terminology (continued)

Term	Definition
security level	A type of security algorithm that is performed on each SNMP packet. There are three levels: noauth, auth, and priv. The noauth level authenticates a packet by a string match of the username. The auth level authenticates a packet by using either the HMAC MD5 or SHA algorithms. The priv level authenticates a packet by using either the HMAC MD5 or SHA algorithms and encrypts the packet using the CBC-DES (DES-56) algorithm.
security model	The security strategy used by the SNMP agent. Currently, Cisco IOS software supports three security models: SNMPv1, SNMPv2c, and SNMPv3.
Simple Network Management Protocol (SNMP)	A network management protocol that provides a method to monitor and control network devices and to manage configurations, statistics collection, performance, and security.
Simple Network Management Protocol Version 2c (SNMPv2c)	Second version of SNMP. This protocol supports centralized and distributed network management strategies and includes improvements in the structure of management information (SMI), protocol operations, management architecture, and security.
SNMP engine	A copy of SNMP that can reside on the local or remote device.
SNMP entity	Unlike SNMPv1 and SNMPv2c, in SNMPv3 the terms SNMP Agents and SNMP Managers are no longer used. These concepts have been combined and are called an SNMP entity. An SNMP entity is made up of an SNMP engine and SNMP applications.
SNMP group	A collection of SNMP users that belong to a common SNMP list that defines an access policy, in which object identification numbers (OIDs) are both read-accessible and write-accessible. Users belonging to a particular SNMP group inherit all of these attributes defined by the group.
SNMP user	A person for which an SNMP management operation is performed. The user is the person on a remote SNMP engine who receives the inform messages.
SNMP view	A mapping between SNMP objects and the access rights that are available for those objects. An object can have different access rights in each view. Access rights indicate whether the object is accessible by either a community string or a user.
write view	A view name (not to exceed 64 characters) for each group; the view name defines the list of object identifiers (OIDs) that can be created or modified by the users of the group.

Understanding How SNMP Works

SNMP is an application-layer protocol that facilitates the exchange of management information between network devices. SNMP enables network administrators to manage network performance, find and solve network problems, and plan for network growth.

There are three versions of SNMP:

- Version 1 (SNMPv1)—This is the initial implementation of SNMP. Refer to RFC 1157 for a full description of functionality. See the “[Understanding How SNMPv1 and SNMPv2c Work](#)” section on page 38-5 for more information on SNMPv1.
- Version 2 (SNMPv2c)—The second release of SNMP, described in RFC 1902, has additions and enhancements to data types, counter size, and protocol operations. See the “[Understanding How SNMPv1 and SNMPv2c Work](#)” section on page 38-5 for more information on SNMPv2.
- Version 3 (SNMPv3)—This is the most recent version of SNMP and is fully described in RFC 2571, RFC 2572, RFC 2573, RFC 2574, and RFC 2575. The SNMP functionality on the Catalyst enterprise LAN switches for SNMPv1 and SNMPv2c remain intact; however, SNMPv3 has significant enhancements to administration and security. See the “[Understanding How SNMPv3 Works](#)” section on page 38-7 for more information on SNMPv3.

Security Models and Levels

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 determines which security mechanism is employed when handling an SNMP packet. Three security models are available: SNMPv1, SNMPv2c, and SNMPv3. [Table 38-2](#) identifies the combinations of security models and defines the levels for SNMPv1, SNMPv2c, and SNMPv3.

Table 38-2 *SNMP Security 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	MD5 or SHA	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms.
v3	authPriv	MD5 or SHA	DES	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms. Provides DES 56-bit encryption in addition to authentication based on the CBC-DES (DES-56) standard.

Note the following about SNMPv3 objects:

- Each user belongs to a group
- A group defines the access policy for a set of users
- SNMP objects access an access policy for reading, writing, and creating
- A group determines the list of notifications its users can receive
- A group also defines the security model and security level for its users

SNMP ifindex Persistence Feature

The SNMP ifIndex persistence feature is always enabled. With the ifIndex persistence feature, the ifIndex value of the port and VLAN is always retained and used after the following occurrences:

- Switch reboot
- High-availability switchover
- Software upgrade
- Module reset
- Module removal and insertion of the same type of module

For Fast EtherChannel and Gigabit EtherChannel interfaces, the ifIndex value is only retained and used after a high-availability switchover.

Understanding How SNMPv1 and SNMPv2c Work

The components of SNMPv1 and SNMPv2c network management fall into three categories:

- Managed devices (such as a switch)
- SNMP agents and MIBs, including Remote Monitoring (RMON) MIBs, which run on managed devices
- SNMP network management applications, such as CiscoWorks2000, which communicate with agents to get statistics and alerts from the managed devices. See the [“Using CiscoWorks2000” section on page 38-6](#) for more information on CiscoWorks2000.



Note An SNMP management application, together with the computer it runs on, is called a Network Management System (NMS).

Using Managed Devices

Catalyst 6500 series switches are managed devices that support SNMP network management with the following features:

- SNMP traps (see the [“Configuring SNMPv1 and SNMPv2c from the CLI” section on page 38-11](#))
- RMON in the supervisor engine software (see [Chapter 39, “Configuring RMON”](#))
- RMON and RMON2 on an external SwitchProbe device

Using SNMP Agents and MIBs

SNMP network management uses these SNMP agent functions:

- Accessing a MIB variable—This function is initiated by the SNMP agent 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.
- Setting a MIB variable—This function is also initiated by the SNMP agent in response to a message from the NMS. The SNMP agent changes the value of the MIB variable to the value that is requested by the NMS.



Note

For more information about MIBs, refer to <http://www.cisco.com/public/sw-center/netmgmt/cmtk/mibs.shtml>.

- SNMP trap—This function is used to notify an NMS that a significant event has occurred at an agent. When a trap condition occurs, the SNMP agent sends an SNMP trap message to any NMSs that are specified as the trap receivers under the following conditions:
 - When a port or module goes up or down
 - When temperature limitations are exceeded
 - When there are spanning tree topology changes
 - When there are authentication failures
 - When power supply errors occur
- SNMP community strings—SNMP community strings authenticate access to MIB objects and function as embedded passwords:
 - Read-only—Gives read access to all objects in the MIB except the community strings but does not allow write access
 - Read-write—Gives read and write access to all objects in the MIB but does not allow access to the community strings
 - Read-write-all—Gives read and write access to all objects in the MIB including the community strings



Note

The community string definitions on your NMS must match at least one of the three community string definitions on the switch.

Using CiscoWorks2000

CiscoWorks2000 is a family of Web-based and management platform-independent products for managing Cisco enterprise networks and devices. CiscoWorks2000 includes Resource Manager Essentials and CWSI Campus, which allow you to deploy, configure, monitor, manage, and troubleshoot a switched internetwork. For more information, refer to the following publications:

- *Getting Started With Resource Manager Essentials*
- *Getting Started With CWSI Campus*

Understanding How SNMPv3 Works

SNMPv3 contains all the functionality of SNMPv1 and SNMPv2c, but SNMPv3 has significant enhancements to administration and security. SNMPv3 is an interoperable standards-based protocol that provides secure access to devices by authenticating and encrypting packets over the network. The security features that are provided in SNMPv3 are as follows:

- Message integrity—Collects data securely without being tampered with or corrupted
- Authentication—Determines that the message is from a valid source
- Encryption—Scrambles the contents of a packet to prevent it from being seen by an unauthorized source

SNMP Entity

Unlike SNMPv1 and SNMPv2c, in SNMPv3 the concept of *SNMP Agents* and *SNMP Managers* no longer apply. These concepts have been combined into an *SNMP entity*. An SNMP entity consists of an SNMP engine and SNMP applications. An SNMP engine consists of the following four components:

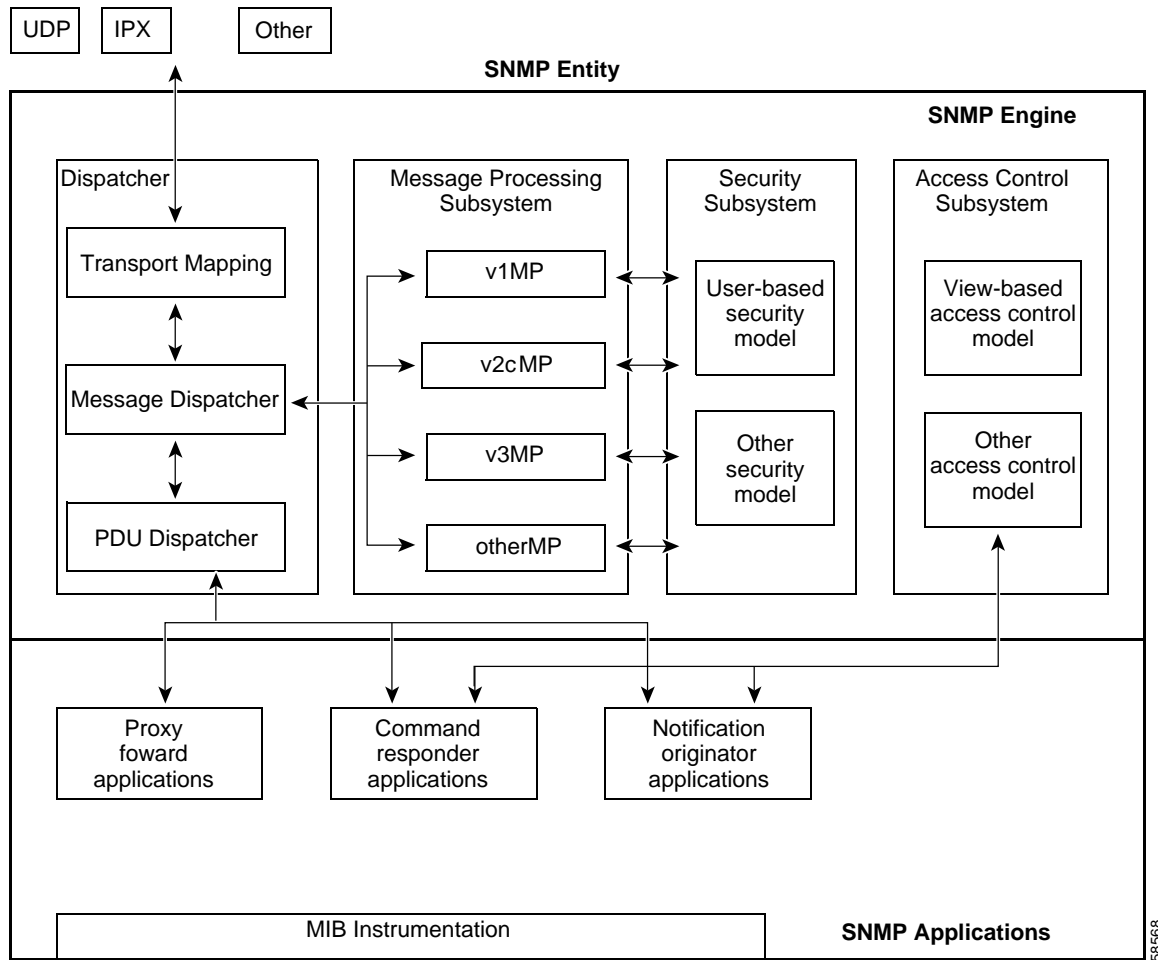
- Dispatcher
- Message processing subsystem
- Security subsystem
- Access control subsystem

Figure 38-1 shows an SNMP entity.

Dispatcher

The dispatcher is a traffic manager that sends and receives messages. After receiving a message, the dispatcher tries to determine the version number of the message and then passes the message to the appropriate message processing model. The dispatcher is also responsible for dispatching protocol data units (PDUs) to applications and for selecting the appropriate transports for sending messages.

Figure 38-1 SNMP Entity for Traditional SNMP Agents



Message Processing Subsystem

The message processing subsystem accepts outgoing PDUs from the dispatcher and prepares them for transmission by wrapping them in a message header and returning them to the dispatcher. The message processing subsystem also accepts incoming messages from the dispatcher, processes each message header, and returns the enclosed PDU to the dispatcher. An implementation of the message processing subsystem may support a single message format corresponding to a single version of SNMP (SNMPv1, SNMPv2c, SNMPv3), or it may contain a number of modules, each supporting a different version of SNMP.

Security Subsystem

The security subsystem authenticates and encrypts messages. Each outgoing message is passed to the security subsystem from the message processing subsystem. Depending on the services required, the security subsystem may encrypt the enclosed PDU and some fields in the message header. In addition, the security subsystem may generate an authentication code and insert it into the message header. After encryption, the message is returned to the message processing subsystem.

Each incoming message is passed to the security subsystem from the message processing subsystem. If required, the security subsystem checks the authentication code and performs decryption. The processed message is returned to the message processing subsystem. An implementation of the security subsystem may support one or more distinct security models. The only currently defined security model is the user-based security model (USM) for SNMPv3, which is specified in RFC 2274.

The USM protects SNMPv3 messages from the following potential security threats:

- An authorized user sending a message that gets modified in transit by an unauthorized SNMP entity.
- An unauthorized user trying to masquerade as an authorized user.
- A user modifying the message stream.
- An unauthorized user listening to the message.

The USM currently defines the HMAC-MD5-96 and HMAC-SHA-96 as the authentication protocols and CBC-DES as the privacy protocol.

SNMPv1 and SNMPv2c security models provide only community names for authentication and no privacy.

Access Control Subsystem

The access control subsystem determines whether access to a managed object should be allowed. With the view-based access control model (VACM), you can control which users and which operations can have access to which managed objects.

Applications

SNMPv3 applications refer to internal applications within an SNMP entity. These internal applications can do the following operations:

- Generate SNMP messages
- Respond to received SNMP messages
- Generate and receive notifications
- Forward messages between SNMP entities

There are currently five types of applications:

- Command generators—Generate SNMP commands to collect or set management data.
- Command responders—Provide access to management data. For example, **processing get**, **get-next**, **get-bulk**, and **set pdus** are used in a command responder application.
- Notification originators—Initiate Trap or Inform messages.
- Notification receivers—Receive and process Trap or Inform messages.
- Proxy forwarders—Forward messages between SNMP entities.

Enabling and Disabling SNMP Processing

This section describes how to use the **set snmp enable | disable** command to enable or disable the processing of SNMP requests to the switch and SNMP traps from the switch.

If you set SNMP to enable mode, SNMP requests to the switch are processed and SNMP traps are sent out if there is no conflict with other SNMP configurations on the switch.

If you set SNMP to disable mode, SNMP requests are ignored and no SNMP traps are sent out independent of the other SNMP configurations on the switch.

In either SNMP mode (enabled or disabled), you can change other SNMP configurations. RMON-related processes are not affected in either mode.

To enable SNMP processing from the command-line interface (CLI), perform this task in privileged mode (enable mode is the default):

	Task	Command
Step 1	Enable SNMP processing.	set snmp enable disable
Step 2	Verify that SNMP processing is enabled.	show snmp

This example shows how to enable SNMP processing:

```
Console> (enable) set snmp enable
SNMP enabled.
Console> (enable)
```

This example shows how to disable SNMP processing:

```
Console> (enable) set snmp disable
SNMP disabled.
Console> (enable)
```

This example shows how to verify the SNMP configuration:

```
Console> (enable) show snmp
SNMP:                Disabled
RMON:                Disabled
Extended RMON Netflow Enabled : None.
Memory usage limit for new RMON entries: 85 percent
Traps Enabled:
None
Port Traps Enabled: None

Community-Access      Community-String
-----
read-only             public
read-write            private
read-write-all       secret

Trap-Rec-Address Trap-Rec-Community Trap-Rec-Port Trap-Rec-Owner Trap-Rec-Index
-----
Console> (enable)
```

Configuring SNMPv1 and SNMPv2c on the Switch

This section provides basic SNMPv1 and SNMPv2c configuration information. For detailed information on the SNMP commands that are supported by the Catalyst 6500 series switches, refer to the *Catalyst 6500 Series Switch Command Reference* publication.

SNMPv1 and SNMPv2c Default Configuration

Refer to the *Catalyst 6500 Series Switch Command Reference* publication for SNMP default configuration settings for each command listed in the configuration section.

Configuring SNMPv1 and SNMPv2c from an NMS

To configure SNMP from an NMS, refer to the NMS documentation (see the [“Using CiscoWorks2000” section on page 38-6](#)).

The switch supports up to 20 trap receivers through the RMON2 trap destination table. You configure the RMON2 trap destination table from the NMS.

Configuring SNMPv1 and SNMPv2c from the CLI



Note

For enhanced SNMP features in software release 7.5(1), see the [“SNMPv1 and SNMPv2c Enhancements in Software Release 7.5\(1\)” section on page 38-12](#).

To configure SNMP from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Define the SNMP community strings for each access type.	set snmp community read-only <i>community_string</i> set snmp community read-write <i>community_string</i> set snmp community read-write-all <i>community_string</i>
Step 2	Assign a trap receiver and community. You can specify up to ten trap receivers.	set snmp trap rcvr_address rcvr_community
Step 3	Specify the SNMP traps to send to the trap receiver.	set snmp trap enable [all auth bridge chassis config entity entityfru envfan envpower envshutdown envtemp flashinsert flashremove ippermit module stpx syslog system vlancreate vlandelete vmpps vtp]
Step 4	Verify the SNMP configuration.	show snmp

This example shows how to define community strings, assign a trap receiver, and specify which traps to send to the trap receiver:

```

Console> (enable) set snmp community read-only Everyone
SNMP read-only community string set to 'Everyone'.
Console> (enable) set snmp community read-write Administrators
SNMP read-write community string set to 'Administrators'.
Console> (enable) set snmp community read-write-all Root
SNMP read-write-all community string set to 'Root'.
Console> (enable) set snmp trap 172.16.10.10 read-write
SNMP trap receiver added.
Console> (enable) set snmp trap 172.16.10.20 read-write-all
SNMP trap receiver added.
Console> (enable) set snmp trap enable all
All SNMP traps enabled.
Console> (enable) show snmp
RMON:                               Disabled
Extended RMON:                       Extended RMON module is not present
Traps Enabled:
Port,Module,Chassis,Bridge,Repeater,Vtp,Auth,ippermit,Vmps,config,entity,stp
Port Traps Enabled: 1/1-2,4/1-48,5/1
Community-Access      Community-String
-----
read-only             Everyone
read-write            Administrators
read-write-all       Root
Trap-Rec-Address      Trap-Rec-Community
-----
172.16.10.10         read-write
172.16.10.20         read-write-all
Console> (enable)

```


Note

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).

SNMPv1 and SNMPv2c Enhancements in Software Release 7.5(1)

These sections describe the enhancements that have been added to software release 7.5(1):

- [Setting Multiple SNMP Community Strings, page 38-13](#)
- [Clearing SNMP Community Strings, page 38-14](#)
- [Specifying Access Numbers for Hosts, page 38-14](#)
- [Clearing IP Addresses Associated with Access Numbers, page 38-15](#)
- [Specifying, Displaying, and Clearing an Interface Alias, page 38-16](#)

Setting Multiple SNMP Community Strings

You can set multiple SNMP community strings using the **community-ext** keyword. Community strings that are defined using the **community-ext** keyword cannot be duplicates of existing community strings. When you add a new community string using the **community-ext** keyword, appropriate entries are created in the `vacmAccessTable` (if a view is specified), `snmpCommunityTable`, and the `vacmSecurityToGroup` table.

To set multiple SNMP community strings from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Set multiple SNMP community strings.	set snmp community-ext <i>community_string</i> { read-only read-write read-write-all } [view <i>view_oid</i>] [access <i>access_number</i>]
Step 2	Verify the SNMP configuration.	show snmp

This example shows how to set an additional SNMP community string:

```
Console> (enable) set snmp community-ext public1 read-only
```

```
Community string public1 is created with access type as read-only
```

```
Console> (enable)
```

This example shows how to restrict the community string to an access number:

```
Console> (enable) set snmp community-ext private1 read-write access 2
```

```
Community string private1 is created with access type as read-write access number 2
```

```
Console> (enable)
```

This example shows how to change the access number to the community string:

```
Console> (enable) set snmp community-ext private1 read-write access 3
```

```
Community string private1 is updated with access type as read-write access number 3
```

```
Console> (enable)
```

This example shows how to display the SNMP configuration:

```
Console> (enable) show snmp
```

```
SNMP:Enabled
RMON:Disabled
Extended RMON Netflow Enabled :None.
Memory usage limit for new RMON entries:85 percent
Traps Enabled:None
Port Traps Enabled:None
```

```
Community-Access Community-String
-----
read-only          public
read-write         private
read-write-all    secret
```

```

Additional-          Access-          Access-
Community-String    Type          Number   View
-----
public1             read-only
public2             read-only     1
private1           read-write   2         1.3.6
secret1            read-write-all 500      1.3.6.1.4.1.9.9

Trap-Rec-Address Trap-Rec-Community Trap-Rec-Port Trap-Rec-Owner Trap-Rec-Index
-----
Console> (enable)

```

Clearing SNMP Community Strings

You can clear community strings using the **clear snmp community-ext** *community-string* command. When you use this command to clear a community string, the corresponding entries in the `vacmAccessTable` and `vacmSecurityToGroup` tables are also removed.

To clear an SNMP community string from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Clear an SNMP community string.	clear snmp community-ext <i>community-string</i>
Step 2	Verify the SNMP configuration.	show snmp

This example shows how to clear an SNMP community string:

```

Console> (enable) clear snmp community-ext public1
Community string public1 has been removed
Console> (enable)

```

Specifying Access Numbers for Hosts

You can specify a list of access numbers that are associated with one or more hosts to limit which hosts can use a specific community string to access the system. You can specify more than one IP address that is associated with an access number by separating each IP address with a space. If an existing access number is used, the new IP addresses are appended to the list.

To specify an access number for a host from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Specify an access number for a host.	set snmp access-list <i>access_number IP_address [ipmask maskaddr]</i>
Step 2	Verify the SNMP configuration.	show snmp access-list

These examples show how to specify an access number for a host:

```

Console> (enable) set snmp access-list 1 172.20.60.100
Access number 1 has been created with new IP Address 172.20.60.100

Console> (enable) set snmp access-list 2 172.20.60.100 mask 255.0.0.0
Access number 2 has been created with new IP Address 172.20.60.100 mask 255.0.0.0

```

```

Console> (enable) set snmp access-list 2 172.20.60.7
Access number 2 has been updated with new IP Address 172.20.60.7

Console> (enable) set snmp access-list 2 172.20.60.7 mask 255.255.255.0
Access number 2 has been updated with existing IP Address 172.20.60.7 mask 255.255.255.0
Console> (enable)

```

This example shows how to display the SNMP configuration:

```

Console> (enable) show snmp access-list
Access-Number  IP-Addresses/IP-Mask
-----
1              172.20.60.100/255.0.0.0
              1.1.1.1/-
2              172.20.60.7/-
              2.2.2.2/-
3              2.2.2.2/155.0.0.0
4              1.1.1.1/2.1.2.4
              2.2.2.2/-
              2.2.2.5/-
Console> (enable)

```

Clearing IP Addresses Associated with Access Numbers

To clear IP addresses that are associated with access numbers from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Clear IP addresses that are associated with access numbers.	clear snmp access-list <i>access_number</i> <i>IP_address</i> [[<i>IP_address</i>] ...]
Step 2	Verify the SNMP configuration.	show snmp access-list

These examples show how to clear IP addresses that are associated with access numbers:

```

Console> (enable) clear snmp access-list 101
All IP addresses associated with access-number 101 have been cleared.
Console> (enable)

Console> (enable) clear snmp access-list 2 172.20.60.8
Access number 2 no longer associated with 172.20.60.8
Console> (enable)

```

Specifying, Displaying, and Clearing an Interface Alias

You can specify, display, and clear an interface alias. The length of the alias can be up to 64 characters.



Note

The **set snmp ifalias** command cannot be used in binary configuration mode. You must use the text file configuration mode when entering this command or the ifalias is not saved in NVRAM.

To specify, display, and clear an interface alias, perform this task in privileged mode:

	Task	Command
Step 1	Specify an interface alias.	set snmp ifalias {ifIndex} [ifAlias]
Step 2	Display the interface alias.	show snmp ifalias [ifIndex]
Step 3	Clear the interface alias.	clear snmp ifalias {ifIndex} all

These examples show how to specify, display, and clear an interface alias:

```
Console> (enable) set snmp ifalias 1 Inband port
```

```
ifIndex 1 alias set
Console> (enable)
```

```
Console> (enable) show snmp ifalias 1
ifIndex   ifName      ifAlias
-----
1         sc0         Inband port
Console> (enable)
```

```
Console> (enable) clear snmp ifalias all
Console> (enable)
```

Configuring SNMPv3 on the Switch

This section provides basic SNMPv3 configuration information. For detailed information on the SNMP commands that are supported by the Catalyst 6500 series switches, refer to the *Catalyst 6500 Series Switch Command Reference* publication.

SNMPv3 Default Configuration

Refer to the *Catalyst 6500 Series Switch Command Reference* publication for SNMP default configuration settings for each command that is listed in the configuration section.

Configuring SNMPv3 from an NMS

To configure SNMP from an NMS, refer to the NMS documentation (see the [“Using CiscoWorks2000” section on page 38-6](#)).

The switch supports up to 20 trap receivers through the RMON2 trap destination table. You configure the RMON2 trap destination table from the NMS.

Configuring SNMPv3 from the CLI

To configure SNMPv3 from the CLI, perform this task in privileged mode:

	Task	Command
Step 1	Set the SNMP-Server EngineID name for the local SNMP engine.	set snmp engineid <i>engineid</i>
Step 2	Configure the MIB views.	set snmp view [-hex] {viewname} {subtree} [mask] [included excluded] [volatile nonvolatile]
Step 3	Set the access rights for a group with a certain security model in different security levels.	set snmp access [-hex] {groupname} {security-model v3} {noauthentication authentication privacy} [read [-hex] {readview}] [write [-hex] {writeview}] [notify [-hex] {notifyview}] [context [-hex] {contextname} [exact prefix]] [volatile nonvolatile]
Step 4	Specify the target addresses for notifications.	set snmp notify [-hex] {notifyname} tag [-hex] {notifytag} [trap inform] [volatile nonvolatile]
Step 5	Set the snmpTargetAddrEntry in the target address table.	set snmp targetaddr [-hex] {addrname} param [-hex] {paramsname} {ipaddr} [udpport {port}] [timeout {value}] [retries {value}] [volatile nonvolatile] [taglist [{-hex} tag] [{-hex} tag]]
Step 6	Set the SNMP parameters that are used to generate a message to a target.	set snmp targetparams [-hex] {paramsname} user [-hex] {username} {security-model v3} {message-processing v3} {noauthentication authentication privacy} [volatile nonvolatile]
Step 7	Configure a new user.	set snmp user [-hex] {username} [remote {engineid}] [{authentication {md5 sha} {authpassword}}] [privacy {privpassword}] [volatile nonvolatile]
Step 8	Relate a user to a group using a specified security model.	set snmp group [-hex] {groupname} user [-hex] {username} {security-model v1 v2 v3} [volatile nonvolatile]
Step 9	Configure the community table for the system default part, which maps community strings of previous versions of SNMP to SNMPv3.	set snmp community {read-only read-write read-write-all} [community_string]
Step 10	Configure the community table for mappings between different community strings and security models with full permissions.	set snmp community index {index_name} name [community_string] security {security_name} context {context_name} transporttag {tag_value} [volatile nonvolatile]
Step 11	Verify the SNMP configuration.	show snmp

This example shows how to set a MIB view to interfacesMibView:

```
Console> (enable) set snmp view interfacesMibView 1.3.6.1.2.1.2 included
Snmp view name was set to interfacesMibView with subtree 1.3.6.1.2.1.2 included,
nonvolatile.
```

This example shows how to set the access rights for a group called `guestgroup` to SNMPv3 authentication-read mode:

```
Console> (enable) set snmp access guestgroup security-model v3 authentication read
interfacesMibView
Snmp access group was set to guestgroup version v3 level authentication,
readview interfacesMibView, context match:exact, nonvolatile.
```

This example shows how to specify the target addresses:

```
Console> (enable) set snmp notify notifytable1 tag routers trap
Snmp notify name was set to notifytable1 with tag routers notifyType trap, and storageType
nonvolatile.
```

These examples show how to set the `snmpTargetAddrEntry` in the target address table:

```
Console> (enable) set snmp targetaddr router_1 param p1 172.20.21.1
Snmp targetaddr name was set to router_1 with param p1
ipAddr 172.20.21.1, udpport 162, timeout 1500, retries 3, storageType nonvolatile.
```

```
Console> (enable) set snmp targetaddr router_2 param p2 172.20.30.1
Snmp targetaddr name was set to router_2 with param p2
ipAddr 172.20.30.1, udpport 162, timeout 1500, retries 3, storageType nonvolatile.
```

These examples show how to set SNMP target parameters:

```
Console> (enable) set snmp targetparams p1 user guestuser1 security-model v3
message-processing v3 authentication
Snmp target params was set to p1 v3 authentication, message-processing v3,
user guestuser1 nonvolatile.
```

```
Console> (enable) set snmp targetparams p2 user guestuser2 security-model v3
message-processing v3 privacy
Snmp target params was set to p2 v3 privacy, message-processing v3,
user guestuser2 nonvolatile.
```

These examples show how to configure `guestuser1` and `guestuser2` as users:

```
Console> (enable) set snmp user guestuser1 authentication md5 guestuser1password privacy
privacypasswd1
Snmp user was set to guestuser1 authProt md5 authPasswd guestuser1password privProt des
privPasswd
privacypasswd1 with engineid 00:00:00:09:00:10:7b:f2:82:00:00:00 nonvolatile.
```

```
Console> (enable) set snmp user guestuser2 authentication sha guestuser2password
Snmp user was set to guestuser2 authProt sha authPasswd guestuser2password privProt
no-priv with engineid
00:00:00:09:00:10:7b:f2:82:00:00:00 nonvolatile.
```

These examples show how to set `guestuser1` and `guestuser2` as members of the groups `guestgroup` and `mygroup`:

```
Console> (enable) set snmp group guestgroup user guestuser1 security-model v3
Snmp group was set to guestgroup user guestuser1 and version v3, nonvolatile.
```

```
Console> (enable) set snmp group mygroup user guestuser1 security-model v3
Snmp group was set to mygroup user guestuser1 and version v3, nonvolatile.
```

```
Console> (enable) set snmp group mygroup user guestuser2 security-model v3
Snmp group was set to mygroup user guestuser2 and version v3, nonvolatile.
```

This example shows how to verify the SNMPv3 setup for guestuser1 from a workstation:

```
workstation% getnext -v3 10.6.4.201 guestuser1 ifDescr.0
Enter Authentication password :guestuser1password
Enter Privacy password      :privacypasswd1
ifDescr.1 = sc0
```

This example shows how to verify the SNMPv3 setup for guestgroup in the snmpEngineID MIB from a workstation:

```
workstation% getnext -v3 10.6.4.201 guestuser1 snmpEngineID
Enter Authentication password :guestuser1password
Enter Privacy password      :privacypasswd1
snmpEngineID = END_OF_MIB_VIEW_EXCEPTION
```

This example shows how to verify the SNMPv2c setup for public access from a workstation:

```
workstation% getnext -v2c 10.6.4.201 public snmpEngineID
snmpEngineID.0 =
00 00 00 09 00 10 7b f2 82 00 00 00
```

These examples show how to increase guestgroup's access right to read privileges for snmpEngineMibView:

```
Console> (enable) set snmp view snmpEngineMibView 1.3.6.1.6.3.10.2.1 included
Snmp view name was set to snmpEngineMibView with subtree 1.3.6.1.6.3.10.2.1 included,
nonvolatile
```

```
Console> (enable) set snmp access guestgroup security-model v3 authentication read
snmpEngineMibView
Snmp access group was set to guestgroup version v3 level authentication,
readview snmpEngineMibView, nonvolatile.
```

This example shows how to verify the SNMPv3 access for guestuser1 from a workstation:

```
workstation% getnext -v3 10.6.4.201 guestuser1 snmpEngineID
Enter Authentication password :guestuser1password
Enter Privacy password      :privacypasswd1
snmpEngineID.0 =
00 00 00 09 00 10 7b f2 82 00 00 00
```

This example shows how to remove access for guestgroup:

```
Console> (enable) clear snmp acc guestgroup security-model v3 authentication
Cleared snmp access guestgroup version v3 level authentication.
```

This example shows how to verify that the access for guestuser1 has been removed from a workstation:

```
workstation% getnext -v3 10.6.4.201 guestuser1 ifDescr.1
Enter Authentication password :guestuser1password
Enter Privacy password      :privacypasswd1
Error code set in packet - AUTHORIZATION_ERROR:1.
```

This example shows how to verify the access for guestuser2 from a workstation:

```
workstation% getnext -v3 10.6.4.201 guestuser2 ifDescr.1
Enter Authentication password :guestuser2password
Enter Privacy password      :privacypasswd2
REPORT received, cannot recover:
usmStatsUnsupportedSecLevels.0 = 1
```

