SNMP Configuration Guide, Cisco IOS Release 12.4T

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

CONTENTS

Periodic MIB Data Collection and Transfer Mechanism

This document describes how to periodically transfer selected MIB data from Cisco IOS-based devices to specified Network Management Systems (NMS).

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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

Prerequisites for Periodic MIB Data Collection and Transfer Mechanism

To use this feature, you should be familiar with the Simple Network Management Protocol (SNMP) model of management information. You should also know what MIB information you want to monitor on your network devices, and the OIDs or object names for the MIB objects to be monitored.

Restrictions for Periodic MIB Data Collection and Transfer Mechanism

Cisco Data Collection MIB configuration using SNMP is not currently implemented.

For specific restrictions, see the tasks in the How to Configure Periodic MIB Data Collection and Transfer Mechanism, page 3.

Information About Periodic MIB Data Collection and Transfer Mechanism

Note In the Cisco IOS CLI, the Periodic MIB Data Collection and Transfer Mechanism is referred to as the Bulk Statistics feature.

SNMP Objects and Instances

A type (or class) of SNMP management information is called an object. A specific instance from a type of management information is called an object instance (or SNMP variable). To configure a bulk statistics collection, you must specify the object types to be monitored using a bulk statistics object list and the specific instances of those objects to be collected using a bulk statistics schema.

MIBs, MIB tables, MIB objects, and object indices can all be specified using a series of numbers called an object identifier (OID). OIDs are used in configuring a bulk statistics collection in both the bulk statistics object lists (for general objects) and in the bulk statistics schemas (for specific object instances).

Bulk Statistics Object Lists

To group the MIB objects to be polled, you will need to create one or more object lists. A bulk statistics object list is a user-specified set of MIB objects that share the same MIB index. Object lists are identified using a name that you specify. Named bulk statistics object lists allow the same configuration to be reused in different bulk statistics schemas.

All the objects in an object list must share the same MIB index. However, the objects do not need to be in the same MIB and do not need to belong to the same MIB table. For example, it is possible to group ifInOctets and an Ethernet MIB object in the same schema, because the containing tables for both objects are indexed by the ifIndex.

Bulk Statistics Schemas

Data selection for the Periodic MIB Data Collection and Transfer Mechanism requires the definition of a schema with the following information:

- Name of an object list.
- Instance (specific or wildcarded) that needs to be retrieved for objects in above object list.
- How often the specified instances need to be sampled (polling interval).

A bulk statistics schema is also identified using a name that you specify. This name is used when configuring the transfer options.

Bulk Statistics Transfer Options

After configuring the data to be collected, a single virtual file (VFile or "bulk statistics file") with all collected data is created. This file can be transferred to a network management station (NMS) using FTP, rcp, or TFTP. You can specify how often this file should be transferred. The default transfer interval is once every 30 minutes. You can also configure a secondary destination for the file to be used if, for whatever reason, the file cannot be transferred to the primary network management station.

The value of the transfer interval is also the collection period (collection interval) for the local bulk statistics file. After the collection period ends, the bulk statistics file is frozen, and a new local bulk statistics file is created for storing data. The frozen bulk statistics file is then transferred to the specified destination.

By default, the local bulk statistics file is deleted after successful transfer to an NMS. However, you can configure the routing device to keep the bulk statistics file in memory for a specified amount of time.

An SNMP notification (trap) can be sent to the NMS if a transfer to the primary or secondary NMS is not successful. Additionally, a syslog message will be logged on the local device if transfers are unsuccessful.

Benefits of the Periodic MIB Data Collection and Transfer Mechanism

The Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature) allows many of the same functions as the Bulk File MIB (CISCO-BULK-FILE-MIB.my), but offers some key advantages.

The main advantage is that this feature can be configured through the CLI and does not require an external monitoring application.

The Periodic MIB Data Collection and Transfer Mechanism is mainly targeted for medium to high-end platforms that have sufficient local storage (volatile or permanent) to store bulk statistics files. Locally storing bulk statistics files helps minimize loss of data during temporary network outages.

This feature also has more powerful data selection features than the Bulkfile MIB; it allows grouping of MIB objects from different tables into data groups (object lists). It also incorporates a more flexible instance selection mechanism, where the application is not restricted to fetching an entire MIB table.

How to Configure Periodic MIB Data Collection and Transfer Mechanism

- Configuring a Bulk Statistics Object List, page 4
- Configuring a Bulk Statistics Schema, page 5

Configuring a Bulk Statistics Object List

The first step in configuring the Periodic MIB Data Collection and Transfer Mechanism is to configure one or more object lists.

Note All the objects in a bulk statistics object list have to be indexed by the same MIB index. However, the objects in the object list do not need to belong to the same MIB or MIB table.

When specifying an object name instead of an OID (using the **add** command), only object names from the Interfaces MIB (IF-MIB.my), Cisco Committed Access Rate MIB (CISCO-CAR-MIB.my) and the MPLS Traffic Engineering MIB (MPLS-TE-MIB.my) may be used.

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib bulkstat object-list** *list-name*
4. **add** {*oid* | *object-name*}
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib bulkstat object-list** *list-name* **Example:** `Router(config)# snmp mib bulkstat object-list ifMib`	Defines an SNMP bulk statistics object list and enters Bulk Statistics Object List configuration mode.

	Command or Action	Purpose

Step 4	**add** {*oid* | *object-name*}
	
	Example:
	`Router(config-bulk-objects)# add 1.3.6.1.2.1.2.2.1.11`
	
	Example:
	`Router(config-bulk-objects)# add ifAdminStatus`
	
	Example:
	`Router(config-bulk-objects)# add ifDescr`
	
	Example:
	.
	
	Example:
	.
	
	Example:
	.
		Adds a MIB object to the bulk statistics object list.
		• Repeat as desired until all objects to be monitored in this list are added.

Step 5	**exit** **Example:** `Router(config-bulk-objects)# exit`	Exits from Bulk Statistics Object List configuration mode.

Configuring a Bulk Statistics Schema

The next step in configuring the Periodic MIB Data Collection and Transfer Mechanism is to configure one or more schemas.

The bulk statistics object list to be used in the schema must be defined.

Note Only one object list can be associated with a schema at a time.

>

SUMMARY STEPS

1. **snmp mib bulkstat schema** *schema-name*
2. **object-list** *list-name*
3. Do one of the following:
   - **instance** {**exact** | **wild**} {**interface** *interface-id* [**sub-if**] | **controller** *controller-id* [**sub-if**] | **oid** *oid*}
4. **instance range start** *oid* **end** *oid*
5. **instance repetition** *oid* - *instance* **max** *repeat-number*
6. **poll-interval** *minutes*
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**snmp mib bulkstat schema** *schema-name* **Example:** `Router(config)# snmp mib bulkstat schema intE0`	Names the bulk statistics schema and enters Bulk Statistics Schema (config-bulk-sc) configuration mode.
Step 2	**object-list** *list-name* **Example:** `Router(config-bulk-sc)# object-list ifMib`	Specifies the bulk statistics object list to be included in this schema. Specify only one object list per schema. (If multiple **object-list**commands are executed, the earlier ones are overwritten by newer commands.)

Command or Action	Purpose

Step 3 Do one of the following:
• **instance** {**exact** | **wild**} {**interface** *interface-id* [**sub-if**] | **controller** *controller-id* [**sub-if**] | **oid** *oid*}
Example:
`Router(config-bulk-sc)# instance wild oid 1`
Example:
Example:
`Router(config-bulk-sc)# instance exact interface FastEthernet 0/1 subif`
	Specifies the instance information for objects in this schema.
	• The **instance exact**command indicates that the specified instance, when appended to the object list, is the complete OID.
	• The **instance wild**command indicates that all subindices of the specified OID belong to this schema. The **wild** keyword allows you to specify a partial, "wild carded" instance.
	• Instead of specifying an instance OID, you can specify a specific interface. The **interface** *interface-id* syntax allows you to specify an interface name and number (for example, interface Ethernet 0) instead of specifying the ifIndex OID for the interface. Similarly, the **controller** *controller-id* syntax allows you to specify a controller card (interface). This option is platform dependent.
	• The optional **sub-if** keyword, when added after specifying an interface or controller, includes the ifIndexes for all subinterfaces of the interface you specified.
	• Only one **instance** command can be configured per schema. (If multiple instance commands are executed, the earlier ones are overwritten by new commands.)

Step 4 **instance range start** *oid* **end** *oid* **Example:** `instance range start 1 end 2`	(Optional) When used in conjunction with the **snmp mib bulkstat schema** command, the **instance range** command can be used to configure a range of instances on which to collect data.
Step 5 **instance repetition** *oid* **-** *instance* **max** *repeat-number* **Example:** `instance repetition 1 max 4`	*(Optional)* When used in conjunction with the **snmp mib bulkstat schema** command, the **instance repetition** command can be used to configure data collection to repeat for a certain number of instances of a MIB object.
Step 6 **poll-interval** *minutes* **Example:** `Router(config-bulk-sc)# poll-interval 10`	Sets how often data should be collected from the object instances specified in this schema, in minutes. The default is once every 5 minutes. The valid range is from 1 to 20000.
Step 7 **exit** **Example:** `Router(config-bulk-objects)# exit`	Exits from Bulk Statistics Schema configuration mode.

Configuring a Bulk Statistics Transfer Options

The final step in configuring the Periodic MIB Data Collection and Transfer Mechanism is to configure the transfer options. The collected MIB data are kept in a local file-like entity called a VFile (virtual file, referred to as a bulk statistics file in this document). This file can be transferred to a remote network management station (NMS) at intervals you specify.

The bulk statistics object lists and bulk statistics schemas should be defined before configuring the bulk statistics transfer options.

Note Transfers can only be performed using schemaASCII (cdcSchemaASCII) format. SchemaASCII is an ASCII format that contains parser-friendly hints for parsing data values.

>

SUMMARY STEPS

1. **snmp mib bulkstat transfer** *transfer-id*
2. **buffer-size** *bytes*
3. **format** {**bulkBinary** | **bulkASCII** | **schemaASCII**}
4. **schema** *schema-name*
5. **transfer-interval** *minutes*
6. **url primary** *url*
7. **url secondary** *url*
8. **retry** *number*
9. **retain** *minutes*
10. **enable**
11. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**snmp mib bulkstat transfer** *transfer-id* **Example:** `Router(config)# snmp mib bulkstat transfer bulkstat1`	Identifies the transfer configuration with a name (*transfer-id*) and enters Bulk Statistics Transfer configuration mode.
Step 2	**buffer-size** *bytes* **Example:** `Router(config-bulk-tr)# buffer-size 3072`	(Optional) Specifies the maximum size for the bulk statistics data file, in bytes. The valid range is from 1024 to 2147483647 bytes. The default buffer size is 2048 bytes. **Note** A configurable buffer size limit is available only as a safety feature. Normal bulk statistics files should not generally meet or exceed the default value.

	Command or Action	Purpose
Step 3	**format** {**bulkBinary** | **bulkASCII** | **schemaASCII**} **Example:** `Router(config-bulk-tr)# format schemaASCII`	(Optional) Specifies the format of the bulk statistics data file (VFile). The default is schemaASCII. **Note** Transfers can only be performed using schemaASCII (cdcSchemaASCII) format. SchemaASCII is a human-readable format that contains parser-friendly hints for parsing data values.

Step 4	**schema** *schema-name*
	
	Example:
	`Router(config-bulk-tr)# schema ATM2/0-IFMIB`
	
	Example:
	`Router(config-bulk-tr)# schema ATM2/0-CAR`
	
	Example:
	`Router(config-bulk-tr)# schema Ethernet2/1-IFMIB`
	
	Example:
	.
	
	Example:
	.
	
	Example:
	.
		Specifies the bulk statistics schema to be transferred. Repeat this command as desired. Multiple schemas can be associated with a single transfer configuration; all collected data will be in a single bulk data file (VFile).

Step 5	**transfer-interval** *minutes* **Example:** `Router(config-bulk-tr)# transfer-interval 20`	(Optional) Specifies how often the bulk statistics file should be transferred, in minutes. The default value is once every 30 minutes. The transfer interval is the same as the collection interval.

	Command or Action	Purpose
Step 6	**url primary** *url* **Example:** `Router(config-bulk-tr)# url primary ftp://user:password@host/folder/bulkstat1`	Specifies the network management system (host) that the bulk statistics data file should be transferred to, and the protocol to use for transfer. The destination is specified as a Uniform Resource Locator (URL). • FTP, rcp, or TFTP can be used for the bulk statistics file transfer.
Step 7	**url secondary** *url* **Example:** `Router(config-bulk-tr)# url secondary tftp://10.1.0.1/tftpboot/user/bulkstat1`	(Optional) Specifies a backup transfer destination and protocol for use in the event that transfer to the primary location fails. • FTP, rcp, or TFTP can be used for the bulk statistics file transfer.
Step 8	**retry** *number* **Example:** `Router(config-bulk-tr)# retry 1`	(Optional) Specifies the number of transmission retries. The default value is 0 (in other words, no retries). • If an attempt to send the bulk statistics file fails, the system can be configured to attempt to send the file again using this command. One retry includes an attempt first to the primary destination then, if the transmission fails, to the secondary location; for example, if the retry value is 1, an attempt will be made first to the primary URL, then to the secondary URL, then to the primary URL again, then to the secondary URL again. • The valid range is from 0 to 100.
Step 9	**retain** *minutes* **Example:** `Router(config-bulk-tr)# retain 60`	(Optional) Specifies how long the bulk statistics file should be kept in system memory, in minutes, after the completion of the collection interval and a transmission attempt is made. The default value is 0. • Zero (0) indicates that the file will be deleted immediately after a successful transfer. **Note** If the **retry** command is used, you should configure a retain interval larger than 0. The interval between retries is the retain interval divided by the retry number. For example, if **retain 10** and **retry 2** are configured, retries will be attempted once every 5 minutes. Therefore, if retain 0 is configured, no retries will be attempted. • The valid range is from 0 to 20000.

	Command or Action	Purpose
Step 10	**enable** **Example:** `Router(config-bulk-tr)# enable`	Begins the bulk statistics data collection and transfer process for this configuration. • For successful execution of this action, at least one schema with non-zero number of objects should be configured. • Periodic collection and file transfer operations will commence only if this command is configured. Conversely, the **no enable** command will stop the collection process. A subsequent **enable** will start the operations again. • Each time the collection process is started using the **enable** command, data is collected into a new bulk statistics file. When the **no enable** command is used, the transfer process for any collected data will immediately begin (in other words, the existing bulk statistics file will be transferred to the specified management station).
Step 11	**exit** **Example:** `Router(config-bulk-tr)# exit`	Exits from Bulk Statistics Transfer configuration mode.

• Troubleshooting Tips, page 11

Troubleshooting Tips

If the maximum buffer size for a bulk statistics file is reached before the transfer interval time expires, the transfer operation will still be initiated, and bulk statistics data will be collected into a new file in the system buffer. To correct this behavior, you can decrease the polling frequency, or increase the size of the bulk statistics buffer. If **retain 0** is configured, no retries will be attempted. This is because the interval between retries is the retain value divided by the retry value. For example, if **retain 10** and **retry 2** are configured, retries will be attempted once every 5 minutes. Therefore, if you configure the **retry** command, you should also configure an appropriate value for the **retain** command.

Enabling Monitoring for Bulk Statistics Collection

Optionally, you can enable SNMP notifications to be sent, which provide information on the transfer status of the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature).

SUMMARY STEPS

1. **configure terminal**
2. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [*acl-number*]
3. **snmp-server enable traps bulkstat** [**collection** | **transfer**]
4. **snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [**bulkstat**]
5. **do copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 2	**snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [*acl-number*] **Example:** `Router(config)# snmp-server community public`	Specifies the SNMP community and access options for the device.
Step 3	**snmp-server enable traps bulkstat** [**collection** | **transfer**] **Example:** `Router(config)# snmp-server enable traps bulkstat`	Enables the sending of bulk statistics SNMP notifications (traps or informs). The following notifications (defined in the CISCO-DATA-COLLECTION-MIB) are enabled with this command: • transfer (cdcFileXferComplete)--Sent when a transfer attempt is successful and when a transfer attempt fails. (The varbind cdcFilXferStatus object in the trap defines tells if the transfer is successful or not). • collection (cdcVFileCollectionError)--Sent when data collection could not be carried out successfully.One possible reason for this condition could be insufficient memory on the device to carry out data collection.
Step 4	**snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [**bulkstat**] **Example:** `Router(config)# snmp-server host informs public bulkstat`	Specifies the recipient (host) for the SNMP notifications, and additional transfer options.

	Command or Action	Purpose
Step 5	**do copy running-config startup-config** **Example:** **Example:** `Router(config)# do copy running-config startup-config`	(Optional) Saves the current configuration to NVRAM as the startup configuration file. • The **do** command allows you to execute EXEC mode commands in any configuration mode.

Monitoring and Troubleshooting Periodic MIB Data Collection and Transfer Mechanism

The **show** command for this feature displays the status of the bulk statistics processes. The **debug** command enables the standard set of debugging messages for technical support purposes.

SUMMARY STEPS

1. **show snmp mib bulkstat transfer** [*transfer-name*]
2. **debug snmp bulkstat**

DETAILED STEPS

Command or Action	Purpose

Step 1 **show snmp mib bulkstat transfer** [*transfer-name*]
 	
Example:
`Router# show snmp mib bulkstat transfer`
 	
Example:
 	
Example:
`Transfer Name : ifmib`
 	
Example:
`Retained files`
 	
Example:
 	
Example:
`File Name  : Time Left (in seconds)  :STATE`
 	
Example:
`----------`
 	
Example:
`ifmib_Router_020421_100554683 : 173 : Retry (2 Retry attempt(s) Left)`
 	
Example:
 	
Example:
`ifmib_Router_020421_100554683 : 53 : Retained`
	(Optional) The **show** command for this feature lists all bulk statistics virtual files (VFiles) on the system that have finished collecting data. (Data files that are not complete are not displayed.)
	
	The output lists all of the completed local bulk statistics files, the remaining time left before the bulk statistics file is deleted (remaining retention period), and the state of the bulk statistics file.
	
	The "STATE" of the bulk statistics file will be one of the following:
	
	• Queued--Indicates that the data collection for this bulk statistics file is completed (in other words, the transfer interval has been met) and that the bulk statistics file is waiting for transfer to the configured destination(s).
	• Retry--Indicates that one or more transfer attempts have failed and that the file transfer will be attempted again. The number of retry attempts remaining will be displayed in parenthesis.
	• Retained--Indicates that the bulk statistics file has either been successfully transmitted or that the configured number of retries have been completed.
	
	Tip To determine if a transfer was successful, enable the bulk statistics SNMP notification.
	
	To display only the status of a named transfer (as opposed to all configured transfers), specify the name of the transfer in the *transfer-name* argument.

	Command or Action	Purpose
Step 2	**debug snmp bulkstat** **Example:** `Router# debug snmp bulkstat`	(Optional) Enables standard debugging output for the Bulk Statistics feature. Debugging output includes messages about the creation, transfer, and deletion of bulk statistics files.

Configuration Examples for Periodic MIB Data Collection and Transfer Mechanism

Example Configuring Periodic MIB Data Collection and Transfer Mechanism

This section provides a complete example of configuring the Periodic MIB Data Collection and Transfer Mechanism (Bulk Statistics feature). The example is described in the following subsections:

Transfer Parameters

The following transfer parameters are used for the “Configuring the Periodic MIB Data Collection and Transfer Mechanism” example:

- Transfer interval (collection interval)--30 minutes
- Primary URL--ftp://john:pswrd@cbin2-host/users/john/bulkstat1
- Secondary URL--tftp://john@10.1.1.1/tftpboot/john/bulkstat1
- Transfer format--schemaASCII
- Retry interval--Retry after 6 minutes (retry = 5, retain = 30; 5 retry attempts over the 30-minute retention interval.)

Polling Requirements

The following polling requirements for ATM interface 2/0 and Ethernet interface 2/1 are used for the “Configuring the Periodic MIB Data Collection and Transfer Mechanism” example:

ATM interface 2/0

- Objects to be polled--ifInOctets, ifOutOctets, ifInUcastPkts, ifInDiscards, CcarStatSwitchedPkts, CcarStatSwitchedBytes, CcarStatFilteredBytes
- Polling interval--Once every 5 minutes
- Instances--Main interface and all subinterfaces
- For CAR MIB objects, poll all instances related to the specified interface

Ethernet Interface 2/1

- Objects to be polled--ifInOctets, ifOutOctets, ifInUcastPkts, ifInDiscards, CcarStatSwitchedPkts, CcarStatSwitchedBytes, CcarStatFilteredBytes
- Polling interval--Once every 10 minutes
- Instances--Only main interface is to be monitored
- For CAR MIB objects, only include instances pertaining to packets in the incoming direction (on the main interface)

Object List Configuration

Note that since the IF-MIB objects and the CAR-MIB objects do not have the same index, they will have to be a part of different schemas. However, since the objects required are the same for the ATM interface and the Ethernet interface, the object list can be reused for each schema. Therefore, in the following example, an object list is created for the for the IF-MIB objects and another object list is created for the CAR-MIB objects.

```
snmp mib bulkstat object-list ifmib
add ifInoctets
add ifOutoctets
add ifInUcastPkts
add ifInDiscards
exit
snmp mib bulkstat object-list CAR-mib
add CcarStatSwitchedPkts
add CcarStatSwitchedBytes
add CcarStatFilteredBytes
exit
```

Schema Definition Configuration

For the following bulk statistics schema configuration, two schemas are defined for each interface--one for the IF-MIB object instances and one for the CAR-MIB object instances.

```
! ATM IF-MIB schema
snmp mib bulkstat schema ATM2/0-IFMIB
! The following command points to the IF-MIB object list, defined above.
object-list ifmib
poll-interval 5
instance exact interface ATM2/0 subif
exit
! ATM CAR-MIB schema
snmp mib bulkstat schema-def ATM2/0-CAR
object-list CAR-mib
poll-interval 5
instance wildcard interface ATM2/0 subif
exit
!Ethernet IF-MIB schema
snmp mib bulkstat schema Ethernet2/1-IFMIB
object-list ifmib
poll-interval 5
instance exact interface Ethernet2/1
```

```
exit
! Ethernet CAR-MIB schema
snmp mib bulkstat schema Ethernet2/1-CAR
object-list CAR-mib
poll-interval 5
! Note: ifindex of Ethernet2/1 is 3
instance wildcard oid 3.1
exit
```

Transfer Parameter Configuration

For the transfer of the bulk statistics file, the transfer configuration is given the name bulkstat1. All of the four schema definitions are included in the following transfer configuration.

```
snmp mib bulkstat transfer bulkstat1
schema ATM2/0-IFMIB
schema ATM2/0-CAR
schema Ethernet2/1-IFMIB
schema Ethernet2/1-CAR
url primary ftp://username1:pswrd@cbin2-host/users/username1/bulkstat1
url secondary tftp://username1@10.1.0.1/tftpboot/username1/bulkstat1
format schemaASCII
transfer-interval 30
retry 5
buffer-size 1024
retain 30
end
copy running-config startup-config
```

Displaying Status

The following sample output for the **show snmp mib bulkstat transfer** command shows that the initial transfer attempt and the first retry has failed for the newest file, and four additional retry attempts will be made:

```
Router# show snmp mib bulkstat transfer
Transfer Name : bulkstat1
Primary URL ftp://user:XXXXXXXX@192.168.200.162/
Secondary ftp://user:XXXXXXXX@192.168.200.163/
      Retained files

      File Name              : Time Left (in seconds)     : STATE
     ------------------------------------------------------------------------
      bulkstat1_Router_030307_102519739: 1196         :Retry(4 Retry attempt(s) Left)
      bulkstat1_Router_030307_102219739: 1016         :Retained
      bulkstat1_Router_030307_101919739:  836         :Retained
```

The filename for the bulk statistics file is generated with the following extensions to the name you specify in the **url** command:

specified-filename _device-name _date_time-stamp

The device name is the name of the sending device, as specified in the CLI prompt.

The time-stamp format will depend on your system configuration. Typically, the format for the date is YYYYMMDD or YYMMDD. The time stamp uses a 24-hour clock notation, and the format is HHMMSSmmm (where mmm are milliseconds).

In the example above, the files were created on March 7, 2003, at 10:25 a.m., 10:22 a.m., and 10:19 a.m.

Bulk Statistics Output File

The following is sample output as it appears in the bulk statistics file received at the transfer destination. In this output, the name of the bulk statistics file is bulkstat1_Router_20030131_193354234. Also, note that

the schema definition (Schema-def) for the schema Ethernet2/1-IFMIB was added to the file as the configuration was changed (see comment lines indicated by "!").

```
Schema-def  ATM2/0-IFMIB "%u, %s,  %u, %u, %u, %u"
epochtime ifDescr instanceoid ifInOctets ifOutOctets ifInUcastPkts ifInDiscards
Schema-def  ATM2/0-CAR  "%u, %s,  %s, %u, %u, %u, %u "
epochtime ifDescr instanceoid CcarStatSwitchedPkts  ccarStatSwitchedBytes
CcarStatSwitchedPkts  ccarStatSwitchedBytes
Schema-def  Ethernet2/1-IFMIB "%u, %u,  %u, %u, %u, %u"
epochtime ifDescr instanceoid ifInOctets ifOutOctets ifInUcastPkts ifInDiscards
Schema-def  Ethernet2/1-CAR  "%u, %s, %u, %u, %u, %u "
Epochtime instanceoid CcarStatSwitchedPkts  ccarStatSwitchedBytes CcarStatSwitchedPkts
ccarStatSwitchedBytes
Schema-def GLOBAL "%s, %s,  %s, %u, %u, %u, %u"
            hostname data timeofday sysuptime cpu5min cpu1min cpu5sec
ATM2/0-IFMIB: 954417080, ATM2/0, 2, 95678, 23456, 234, 3456
ATM2/0-IFMIB: 954417080, ATM2/0.1, 8, 95458, 54356, 245, 454
ATM2/0-IFMIB: 954417080, ATM2/0.2, 9, 45678, 8756, 934, 36756
ATM2/0-CAR: 954417083, ATM2/0, 2.1.1, 234, 345, 123, 124
ATM2/0-CAR: 954417083, ATM2/0, 2.2.1, 452, 67, 132, 145
ATM2/0-CAR: 954417083, ATM2/0.1, 8.1.1, 224, 765, 324 234
ATM2/0-CAR: 954417083, ATM2/0.1, 8.2.1, 234, 345, 123, 124
ATM2/0-CAR: 954417083, ATM2/0.2, 9.1.1, 234, 345, 123, 124
ATM2/0-CAR: 954417083, ATM2/0.2, 9.2.1, 452, 67, 132, 145
Ethernet2/1-IFMIB: 954417090, Ethernet2/1, 3, 45678, 8756, 934, 36756
Ethernet2/1-CAR: 954417093, 3.1.1, 234, 345, 123, 124
Ethernet2/1-CAR: 954417093, 3.1.2, 134, 475, 155, 187
ATM2/0-IFMIB: 954417100, ATM2/0, 2, 95678, 23456, 234, 3456
ATM2/0-IFMIB: 954417101, ATM2/0.1, 8, 95458, 54356, 245, 454
ATM2/0-IFMIB: 954417102, ATM2/0.2, 9, 45678, 8756, 934, 36756
ATM2/0-CAR: 954417106, ATM2/0, 2.1.1, 234, 345, 123, 124
ATM2/0-CAR: 954417107, ATM2/0, 2.2.1, 452, 67, 132, 145
ATM2/0-CAR: 954417107, ATM2/0.1, 8.1.1, 224, 765, 324 234
ATM2/0-CAR: 954417108, ATM2/0.1, 8.2.1, 234, 345, 123, 124
ATM2/0-CAR: 954417113, ATM2/0.2, 9.1.1, 234, 345, 123, 124
ATM2/0-CAR: 954417114, ATM2/0.2, 9.2.1, 452, 67, 132, 145
! Here the
Schema-def
 for "
Ehternet2/1-IFMIB
" was changed on the originating device.
Schema-def  Ethernet2/1-IFMIB "%u, %u,  %u, %u, %u, %u"
! The object
ifOutDiscards
 has been added to the object list for this schema.

epochtime ifDescr instanceoid ifInOctets ifOutOctets ifInUcastPkts ifInDiscards
            ifOutDiscards
! The following data sample reflects the change in the configuration.
Ethernet2/1-IFMIB: 954417090, Ethernet2/1, 3, 45678, 8756, 934, 36756, 123
Ethernet2/1-CAR: 954417093, 3.1.1, 234, 345, 123, 124
Ethernet2/1-CAR: 954417093, 3.1.2, 134, 475, 155, 187
GLOBAL: Govinda, 20020129, 115131, 78337, 783337, 2%, 0%, 62%
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
SNMP commands	*Cisco IOS Network Management Command Reference*

Related Topic	Document Title
SNMP configuration tasks	"Configuring SNMP Support" module in the *Cisco IOS Network Management Configuration Guide*

MIBs

MIBs	MIBs Link
This feature supports all Cisco implemented MIBs. This feature uses the Cisco Data Collection MIB (CISCO-DATA-COLLECTION-MIB.my) function of reporting errors and statistics during data collection and transfer. The Cisco Data Collection MIB also supports configuring data collection using the CLI, as well as with SNMP.	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
None	--

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Periodic MIB Data Collection and Transfer Mechanism

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software

release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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

Table 1 Feature Information for Periodic MIB Data Collection and Transfer Mechanism

Feature Name	Releases	Feature Information
Periodic MIB Data Collection and Transfer Mechanism	12.0(24)S 12.2(25)S 12.2(33)SB 12.2(33)SRA 12.2(33)SRC 12.2(33)SXH 12.2(50)SY 12.3(2)T 15.0(1)S Cisco IOS XE 3.1.0SG	The Periodic MIB Data Collection and Transfer Mechanism provides the ability to periodically transfer selected MIB data from Cisco IOS-based devices to specified Network Management Systems (NMS). Using the CLI, data from multiple MIBs can be grouped into lists, and a polling interval (frequency of data collection) can be configured. All the MIB objects in a list are periodically polled using this specified interval. The collected data from the lists can then be transferred to a specified NMS at a user-specified transfer interval (frequency of data transfer) using TFTP, rcp, or FTP. The following commands were introduced or modified by this feature: **add (bulkstat object)**, **buffer-size (bulkstat)**, **context (bulkstat)**, **debug snmp bulkstat**, **enable (bulkstat)**, **format (bulkstat)**, **instance (MIB)**, **instance range**, **instance repetition**, **object-list**, **poll-interval**, **retain**, **retry (bulkstat)**, **schema**, **show snmp mib bulkstat transfer**, **snmp mib bulkstat object-list**, **snmp mib bulkstat schema**, **snmp mib bulkstat transfer**, **snmp-server enable traps bulkstat**, **transfer-interval**, **url (bulkstat)**.

Configuring SNMP Support

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 that is used for monitoring and managing devices in a network.

This module discusses how to enable an SNMP agent on a Cisco device and how to control the sending of SNMP notifications from the agent. For information about using SNMP management systems, see the appropriate documentation for your network management system (NMS) application.

For a complete description of the router monitoring commands mentioned in this document, see the Cisco IOS Network Management Command Reference. To locate documentation of other commands that appear in this document, use the *Cisco IOS Command Reference Master Index* or search online.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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

Restrictions for Configuring SNMP Support

Not all Cisco platforms are supported on the features described in this module. Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support.

Information About Configuring SNMP Support

Components of SNMP

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 monitoring and managing devices in a network.

The SNMP framework is made up of three parts:

SNMP Manager

The SNMP manager is a system that controls and monitors the activities of network hosts using SNMP. The most common managing system is an NMS. The term NMS can be applied either to a dedicated device used for network management or to the applications used on such a device. Several network management applications are available for use with SNMP and range from simple CLI applications to applications that use GUIs, such as the CiscoWorks2000 products.

SNMP Agent

The SNMP agent is the software component within a managed device that maintains the data for the device and reports this data, as needed, to managing systems. The agent resides on the routing device (router, access server, or switch). To enable an SNMP agent on a Cisco routing device, you must define the relationship between the manager and the agent.

Note Although it is possible to configure a Cisco router to be an SNMP agent, this practice is not recommended. Commands that an agent needs to control the SNMP process are available through the Cisco IOS CLI without additional configuration.

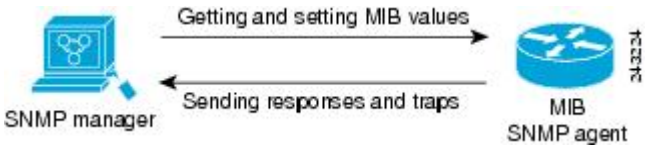
MIB

A MIB is a virtual information storage area for network management information and consists of collections of managed objects. Within a MIB 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 (see the "MIBs and RFCs, page 28" section for an explanation of Request for Comments (RFC) and Standard documents). 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.

An 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 in 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.

The figure below illustrates the communications between the SNMP manager and agent. A manager sends an agent requests to get and set MIB values. The agent responds to these requests. Independent of this interaction, the agent can send the manager unsolicited notifications (traps or informs) to notify the manager about network conditions.

Figure 1 Communication Between an SNMP Agent and Manager



SNMP Operations

SNMP applications perform the following operations to retrieve data, modify SNMP object variables, and send notifications:

- SNMP Get, page 25
- SNMP Set, page 26
- SNMP Notifications, page 26

SNMP Get

The SNMP get operation is performed by an NMS to retrieve SNMP object variables. There are three types of get operations:

- get—Retrieves the exact object instance from the SNMP agent.
- getNext—Retrieves the next object variable, which is a lexicographical successor to the specified variable.
- getBulk—Retrieves a large amount of object variable data, without the need for repeated getNext operations.

SNMP Set

The SNMP set operation is performed by an NMS to modify the value of an object variable.

SNMP Notifications

A key feature of SNMP is its capability to generate unsolicited notifications from an SNMP agent.

- Traps and Informs, page 26

Traps and Informs

Unsolicited (asynchronous) notifications can be generated as traps or inform requests (informs). Traps are messages alerting the SNMP manager to a condition on the network. Informs are traps that include a request for confirmation of receipt from the SNMP manager. Notifications can indicate improper user authentication, restarts, the closing of a connection, loss of connection to a neighbor router, or other significant events.

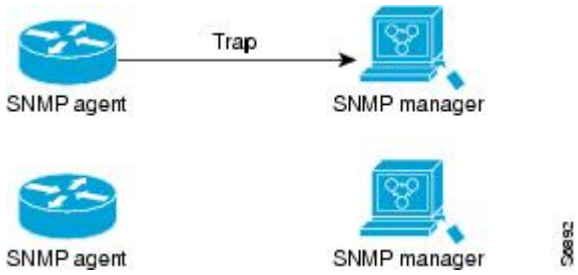
Traps are less reliable than informs because the receiver does not send an acknowledgment when it receives a trap. The sender does not know if the trap was received. An SNMP manager that receives an inform acknowledges the message with an SNMP response protocol data unit (PDU). If the sender never receives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

Traps are often preferred even though they are less reliable because informs consume more resources in the router and the network. Unlike a trap, which is discarded as soon as it is sent, an inform must be held in memory until a response is received or the request times out. Also, traps are sent only once, whereas an inform may be resent several times. The retries increase traffic and contribute to higher overhead on the network. Use of traps and informs requires a trade-off between reliability and resources. If it is important that the SNMP manager receives every notification, use informs. However, if traffic volume or memory usage are concerns and receipt of every notification is not required, use traps.

The figures below illustrate the differences between traps and informs.

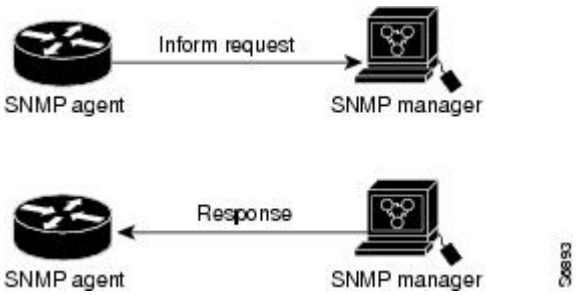
The figure below shows that an agent successfully sends a trap to an SNMP manager. Although the manager receives the trap, it does not send an acknowledgment. The agent has no way of knowing that the trap reached its destination.

Figure 2 Trap Successfully Sent to SNMP Manager



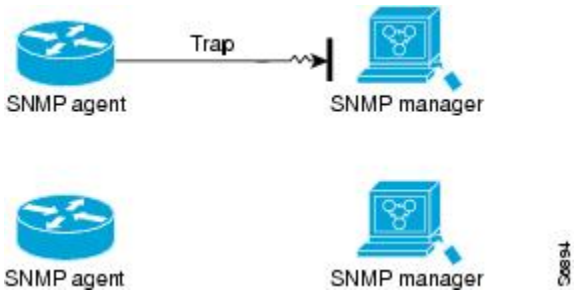
In the figure below, the agent successfully sends an inform to the manager. When the manager receives the inform, a response is sent to the agent, and the agent knows that the inform reached its destination. Note that in this example, the traffic generated is twice as much as in the interaction shown in the figure above.

Figure 3 Inform Request Successfully Sent to SNMP Manager



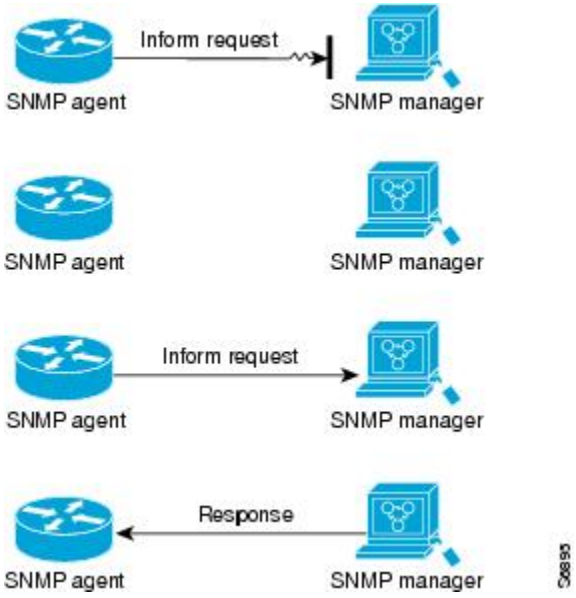
The figure below shows an agent sending a trap to a manager that the manager does not receive. The agent has no way of knowing that the trap did not reach its destination. The manager never receives the trap because traps are not resent.

Figure 4 Trap Unsuccessfully Sent to SNMP Manager



The figure below shows an agent sending an inform to a manager that does not reach the manager. Because the manager did not receive the inform, it does not send a response. After a period of time, the agent resends the inform. The manager receives the inform from the second transmission and replies. In this example, more traffic is generated than in the scenario shown in the figure above, but the notification reaches the SNMP manager.

Figure 5 Inform Unsuccessfully Sent to SNMP Manager



MIBs and RFCs

MIB modules typically are defined in RFC documents submitted to the IETF, an international standards body. RFCs are written by individuals or groups for consideration by the Internet Society and the Internet community as a whole, usually with the intention of establishing a recommended Internet standard. Before being given RFC status, recommendations are published as Internet Draft (I-D) documents. RFCs that have become recommended standards are also labeled as standards documents (STDs). You can learn about the standards process and the activities of the IETF at the Internet Society website at http://www.isoc.org. You can read the full text of all RFCs, I-Ds, and STDs referenced in Cisco documentation at the IETF website at http://www.ietf.org.

The Cisco implementation of SNMP uses the definitions of MIB II variables described in RFC 1213 and definitions of SNMP traps described in RFC 1215.

Cisco provides its own private MIB extensions with every system. Cisco enterprise MIBs comply with the guidelines described in the relevant RFCs unless otherwise noted in the documentation. You can find the MIB module definition files and the list of MIBs supported on each Cisco platform on the Cisco MIB website on Cisco.com.

Versions of SNMP

The Cisco IOS software supports the following versions of SNMP:

- SNMPv1--Simple Network Management Protocol: a full Internet standard, defined in RFC 1157. (RFC 1157 replaces the earlier versions that were published as RFC 1067 and RFC 1098.) Security is based on community strings.
- SNMPv2c--The community string-based Administrative Framework for SNMPv2. SNMPv2c (the "c" is for "community") is an experimental Internet protocol defined in RFC 1901, RFC 1905, and RFC 1906. SNMPv2c is an update of the protocol operations and data types of SNMPv2p (SNMPv2 Classic) and uses the community-based security model of SNMPv1.
- SNMPv3--Version 3 of SNMP. SNMPv3 is an interoperable standards-based protocol defined in RFCs 3413 to 3415. SNMPv3 provides secure access to devices by authenticating and encrypting packets over the network.

The security features provided in SNMPv3 are as follows:

-
  - Message integrity--Ensuring that a packet has not been tampered with in transit.
  - Authentication--Determining that the message is from a valid source.
  - Encryption--Scrambling the contents of a packet to prevent it from being learned by an unauthorized source.

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

SNMPv2c support includes a bulk retrieval mechanism and 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 types of errors; these conditions are reported through a single error code in SNMPv1. The following three types of exceptions are also reported: no such object, no such instance, and end of MIB view.

SNMPv3 is a security model in which an authentication strategy 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. The table below lists the combinations of security models and levels and their meanings.

Table 2 SNMP Security Models and Levels

Model	Level	Authentication	Encryption	What Happens
v1	noAuthNoPriv	Community String	No	Uses a community string match for authentication.
v2c	noAuthNoPriv	Community String	No	Uses a community string match for authentication.
v3	noAuthNoPriv	Username	No	Uses a username match for authentication.
v3	authNoPriv	Message Digest 5 (MD5) or Secure Hash Algorithm (SHA)	No	Provides authentication based on the HMAC-MD5 or HMAC-SHA algorithms.
v3	authPriv	MD5 or SHA	Data Encryption Standard (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 SNMPv2p (SNMPv2 Classic) is not supported in Cisco IOS Release 11.2 and later releases. SNMPv2c replaces the Party-based Administrative and Security Framework of SNMPv2p with a Community-based Administrative Framework. SNMPv2c retained the bulk retrieval and error handling capabilities of SNMPv2p.

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

SNMPv3 supports RFCs 1901 to 1908, 2104, 2206, 2213, 2214, and 2271 to 2275. For additional information about SNMPv3, see RFC 2570, *Introduction to Version 3 of the Internet-standard Network Management Framework* (this document is not a standard).

Cisco-Specific Error Messages for SNMPv3

SNMPv3 provides different levels of security. If an authentication or an authorization request fails, a descriptive error message is returned to indicate what went wrong. These error messages are RFC 3414 compliant.

You can use the **snmp-server usm cisco** command to disable the descriptive messages to prevent malicious users from misusing the information returned in the error messages. The table below lists the error messages returned when the **snmp-server usm cisco** command is used and compares these messages with the corresponding RFC-compliant error messages. The Cisco-specific error messages are a deviation from RFC 3414.

Table 3 Cisco-Specific Error Messages for SNMPv3

Configured Security Level	Security Level of Incoming SNMP Message	RFC 3414-Compliant Error Indication	Cisco-Specific Error Messages
noAuthNoPriv	noAuthNoPriv	No error	No error
	authNoPriv	unsupportedSecurityLevel	unknownUserName
	authPriv	unsupportedSecurityLevel	unknownUserName
authNoPriv	noAuthNoPriv	AUTHORIZATION_ERROR	unknownUserName
	authNoPriv with correct auth password	No error	No error
	authNoPriv with incorrect auth password	wrongDigests	unknownUserName
	authPriv	unsupportedSecurityLevel	unknownUserName
authPriv	noAuthNoPriv	AUTHORIZATION_ERROR	unknownUserName
	authNoPriv with correct auth password	AUTHORIZATION_ERROR	unknownUserName
	authNoPriv with incorrect auth password	AUTHORIZATION_ERROR	unknownUserName
	authPriv with correct auth password and correct priv password	No error	No error

Configured Security Level	Security Level of Incoming SNMP Message	RFC 3414-Compliant Error Indication	Cisco-Specific Error Messages
	authPriv with correct auth password and incorrect priv password	No response	No response
	authPriv with incorrect auth password and incorrect priv password	wrongDigests	unknownUserName
	authPriv with incorrect auth password and correct priv password	wrongDigests	unknownUserName

Note For a configured SNMP user, if the SNMP group is not configured or the group security level is not the same as the user security level, the error returned is AUTHORIZATION_ERROR. The Cisco-specific error message for this scenario is unknownUserName.

Detailed Interface Registration Information

The Interface Index Display for SNMP feature introduces new commands and command modifications that allow advanced users of SNMP to view information about the interface registrations directly on the managed agent. You can display MIB information from the agent without using an external NMS.

Note For the purposes of this document, the agent is a routing device running Cisco IOS software.

This feature addresses three objects in the Interfaces MIB: ifIndex, ifAlias, and ifName. For a complete definition of these objects, see the IF-MIB.my file available from the Cisco SNMPv2 MIB website at ftp://ftp.cisco.com/pub/mibs/v2/.

- Interface Index, page 31
- Interface Alias, page 31
- Interface Name, page 32

Interface Index

The ifIndex object (ifEntry 1) is called the Interface Index. The Interface Index is a unique value greater than zero that identifies each interface or subinterface on the managed device. This value becomes the interface index identification number.

The **show snmp mib ifmib ifindex** command allows you to view SNMP Interface Index Identification numbers assigned to interfaces and subinterfaces. An NMS is not required.

Interface Alias

The ifAlias object (ifXEntry 18) is called the Interface Alias. The Interface Alias is a user-specified description of an interface used for SNMP network management. The ifAlias is an object in the Interfaces

Group MIB (IF-MIB) that can be set by a network manager to "name" an interface. The ifAlias value for an interface or subinterface can be set using the **description** command in interface configuration mode or subinterface configuration mode or by using a Set operation from an NMS. Previously, ifAlias descriptions for subinterfaces were limited to 64 characters. (The OLD-CISCO-INTERFACES-MIB allows up to 255 characters for the locIfDescr MIB variable, but this MIB does not support subinterfaces.) The new **snmp ifmib ifalias long** command configures the system to handle IfAlias descriptions of up to 256 characters. IfAlias descriptions appear in the output of the **show interfaces** command.

Interface Name

The ifName object (ifXEntry 1) is the textual name of the interface. The purpose of the ifName object is to cross reference the CLI representation of a given interface. The value of this object is the name of the interface as assigned by the local device and is suitable for use in CLI commands. If there is no local name or this object is otherwise not applicable, this object contains a zero-length string. No commands introduced by this feature affect the ifName object, but it is discussed here to show its relation to the ifIndex and ifAlias objects.

The **show snmp mib** command shows all objects in the MIB on a Cisco device (similar to a mibwalk). The objects in the MIB tree are sorted using lexical ordering, meaning that object identifiers are sorted in sequential, numerical order. Lexical ordering is important when using the GetNext operation from an NMS because these operations take an object identifier (OID) or a partial OID as input and return the next object from the MIB tree based on the lexical ordering of the tree.

SNMP Support for VPNs

The SNMP Support for VPNs feature allows SNMP traps and informs to be sent and received using VPN routing and forwarding (VRF) tables. In particular, this feature adds support to the Cisco IOS software for sending and receiving SNMP traps and informs specific to individual VPNs.

A VPN is a network that provides high connectivity transfers on a shared system with the same usage guidelines as a private network. A VPN can be built on the Internet over IP, Frame Relay, or ATM networks.

A VRF stores per-VPN routing data. It defines the VPN membership of a customer site attached to the network access server (NAS). A VRF consists of an IP routing table, a derived Cisco Express Forwarding table, and guidelines and routing protocol parameters that control the information that is included in the routing table.

The SNMP Support for VPNs feature provides configuration commands that allow users to associate SNMP agents and managers with specific VRFs. The specified VRF is used for sending SNMP traps and informs and responses between agents and managers. If a VRF is not specified, the default routing table for the VPN is used.

Support for VPNs allows you to configure an SNMP agent to accept only SNMP requests from a certain set of VPNs. With this configuration, service providers can provide network management services to their customers, so customers can manage all user VPN devices.

Interface Index Persistence

One of the identifiers most commonly used in SNMP-based network management applications is the interface index (IfIndex) value . IfIndex is a unique identifying number associated with a physical or logical interface; as far as most software is concerned, the ifIndex is the name of the interface.

Although there is no requirement in the relevant RFCs that the correspondence between particular ifIndex values and their interfaces be maintained across reboots, applications such as device inventory, billing, and fault detection increasingly depend on the maintenance of this correspondence.

This feature adds support for an ifIndex value that can persist across reboots, allowing users to avoid the workarounds previously required for consistent interface identification.

It is currently possible to poll the router at regular intervals to correlate the interfaces to the ifIndex, but it is not practical to poll this interface constantly. If this data is not correlated constantly, however, the data may be made invalid because of a reboot or the insertion of a new card into the router in between polls. Therefore, ifIndex persistence is the only way to guarantee data integrity.

IfIndex persistence means that the mapping between the ifDescr object values and the ifIndex object values (generated from the IF-MIB) will be retained across reboots.

- Benefits of Interface Index Persistence, page 33

Benefits of Interface Index Persistence

- Association of Interfaces with Traffic Targets for Network Management, page 33
- Accuracy for Mediation Fault Detection and Billing, page 33

Association of Interfaces with Traffic Targets for Network Management

The Interface Index Persistence feature allows for greater accuracy when collecting and processing network management data by uniquely identifying input and output interfaces for traffic flows and SNMP statistics. Relating each interface to a known entity (such as an ISP customer) allows network management data to be more effectively utilized.

Accuracy for Mediation Fault Detection and Billing

Network data is increasingly being used worldwide for usage-based billing, network planning, policy enforcement, and trend analysis. The ifIndex information is used to identify input and output interfaces for traffic flows and SNMP statistics. Inability to reliably relate each interface to a known entity, such as a customer, invalidates the data.

MIB Persistence

The MIB Persistence features allow the SNMP data of a MIB to be persistent across reloads; that is, MIB information retains the same set object values each time a networking device reboots. MIB Persistence is enabled by issuing the **snmp mib persist** command, and the MIB data of all MIBs that have had persistence enabled using this command is then written to NVRAM by issuing the **write mib-data** command. All modified MIB data must be written to NVRAM using the **write mib-data** command.

Both Event and Expression MIBs allow you to configure a value for an object and to set up object definitions. Both also allow rows of data to be modified while the row is in an active state.

Scalar objects are stored every time they are changed, and table entries are stored only if the row is in an active state. The Event MIB has two scalar objects and nine tables to be persisted into NVRAM. The tables are as follows:

- mteEventNotificationTable
- mteEventSetTable
- mteEventTable

- mteObjectsTable
- mteTriggerBooleanTable
- mteTriggerDeltaTable
- mteTriggerExistenceTable
- mteTriggerTable
- mteTriggerThresholdTable

The Expression MIB has two scalar objects and three tables to be stored in NVRAM. The scalar objects are expResourceDeltaMinimum and expResourceDeltaWildcardInstanceMaximum. The tables are as follows:

- expExpressionTable
- expNameTable
- expObjectTable

Writing MIB data to NVRAM may take several seconds. The length of time depends on the amount of MIB data.

Event MIB Persistence and Expression MIB Persistence both allow MIB objects to be saved from reboot to reboot, allowing long-term monitoring of specific devices and interfaces and configurations of object values that are preserved across reboots.

Circuit Interface Identification Persistence

The Circuit Interface MIB (CISCO-CIRCUIT-INTERFACE-MIB) provides a MIB object (cciDescr) that can be used to identify individual circuit-based interfaces for SNMP monitoring. The Circuit Interface Identification Persistence for SNMP feature maintains this user-defined name of the circuit across reboots, allowing the consistent identification of circuit interfaces. Circuit Interface Identification Persistence is enabled using the **snmp mib persist circuit** global configuration command.

Cisco IOS Release 12.2(2)T introduces the Circuit Interface Identification Persistence for SNMP feature. The Circuit Interface MIB (CISCO-CIRCUIT-INTERFACE-MIB) provides a MIB object (cciDescr) that can be used to identify individual circuit-based interfaces for SNMP monitoring. The Cisco Circuit Interface MIB was introduced in Cisco IOS Release 12.1(3)T.

The Circuit Interface Identification Persistence for SNMP feature maintains the user-defined name of the circuit (defined in the cciDescr object) across reboots, allowing for the consistent identification of circuits.

The Circuit Interface Identification Persistence for SNMP feature is a supplement to the Interface Index Persistence feature introduced in Cisco IOS Release 12.1(3)T and Cisco IOS Release 12.0(11)S. Circuit Interface Identification Persistence is enabled using the **snmp mib persist circuit** global configuration command. Use this command if you need to consistently identify circuits using SNMP across reboots. This command is disabled by default because this feature uses NVRAM.

In addition, the **show snmp mib ifmib ifindex** EXEC mode command allows you to display the Interfaces MIB ifIndex values directly on your system without an NMS; the **show snmp mib** EXEC mode command allows you to display a list of MIB module identifiers registered directly on your system with an NMS. The **snmp ifmib ifalias long** command allows you to specify a description for interfaces or subinterfaces of up to 256 characters in length. Prior to the introduction of this command, ifAlias descriptions for SNMP management were limited to 64 characters.

Event MIB

The Event MIB provides the ability to monitor MIB objects on a local or remote system using SNMP and initiate simple actions whenever a trigger condition is met; for example, an SNMP trap can be generated

when an object is modified. When the notifications are triggered through events, the NMS does not need to constantly poll managed devices to track changes.

By allowing the SNMP notifications to take place only when a specified condition is met, the Event MIB reduces the load on affected devices and improves the scalability of network management solutions.

The Event MIB operates based on event, object lists configured for the event, event action, trigger, and trigger test.

- Events, page 35
- Object List, page 35
- Trigger, page 35
- Trigger Test, page 35

Events

The event table defines the activities to be performed when an event is triggered. These activities include sending a notification and setting a MIB object. The event table has supplementary tables for additional objects that are configured according to event action. If the event action is set to notification, notifications are sent out whenever the object configured for that event is modified.

Object List

The object table lists objects that can be added to notifications based on trigger, trigger test type, or the event that sends a notification. The Event MIB allows wildcarding, which enables you to monitor multiple instances of an object. To specify a group of object identifiers, you can use the wildcard option.

Trigger

The trigger table defines conditions to trigger events. The trigger table lists the objects to be monitored and associates each trigger with an event. An event occurs when a trigger is activated. To create a trigger, you should configure a trigger entry in the mteTriggerTable of the Event MIB. This trigger entry specifies the object identifier of the object to be monitored. Each trigger is configured to monitor a single object or a group of objects specified by a wildcard (*). The Event MIB process checks the state of the monitored object at specified intervals.

Trigger Test

The trigger table has supplementary tables for additional objects that are configured based on the type of test performed for a trigger. For each trigger entry type such as existence, threshold, or Boolean, the corresponding tables (existence, threshold, and Boolean tables) are populated with the information required to perform the test. The Event MIB allows you to set event triggers based on existence, threshold, and Boolean trigger types. When the specified test on an object returns a value of *true*, the trigger is activated. You can configure the Event MIB to send out notifications to the interested host when a trigger is activated.

Expression MIB

The Expression MIB allows you to create expressions based on a combination of objects. The expressions are evaluated according to the sampling method. The Expression MIB supports the following types of object sampling:

- Absolute

- Delta
- Changed

If there are no delta or change values in an expression, the expression is evaluated when a requester attempts to read the value of the expression. In this case, all requesters get a newly calculated value.

For expressions with delta or change values, an evaluation is performed for every sampling. In this case, requesters get the value as of the last sample period.

- Absolute Sampling, page 36
- Delta Sampling, page 36
- Changed Sampling, page 36

Absolute Sampling

Absolute sampling uses the value of the MIB object during sampling.

Delta Sampling

Delta sampling is used for expressions with counters that are identified based on delta (difference) from one sample to the next. Delta sampling requires the application to do continuous sampling, because it uses the value of the last sample.

Changed Sampling

Changed sampling uses the changed value of the object since the last sample.

SNMP Notification Logging

Systems that support SNMP often need a mechanism for recording notification information. This mechanism protects against notifications being lost because they exceeded retransmission limits. The Notification Log MIB provides a common infrastructure for other MIBs in the form of a local logging function. The SNMP Notification Logging feature adds Cisco IOS CLI commands to change the size of the notification log, to set the global ageout value for the log, and to display logging summaries at the command line. The Notification Log MIB improves notification tracking and provides a central location for tracking all MIBs.

Note The Notification Log MIB supports notification logging on the default log only.

How to Configure SNMP Support

There is no specific command to enable SNMP. The first **snmp-server** command that you enter enables supported versions of SNMP. All other configurations are optional.

- Configuring System Information, page 37
- Configuring SNMP Versions 1 and 2, page 39
- Configuring SNMP Version 3, page 44
- Configuring a Router as an SNMP Manager, page 48
- Enabling the SNMP Agent Shutdown Mechanism, page 51

Configuring System Information

You can set the system contact, location, and serial number of the SNMP agent so that these descriptions can be accessed through the configuration file. Although the configuration steps described in this section are optional, configuring the basic information is recommended because it may be useful when troubleshooting your configuration. In addition, the first **snmp-server** command that you issue enables SNMP on the device.

Perform this task as needed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server contact** *text*
4. **snmp-server location** *text*
5. **snmp-server chassis-id** *number*
6. **exit**
7. **show snmp contact**
8. **show snmp location**
9. **show snmp chassis**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server contact** *text* **Example:** `Router(config)# snmp-server contact NameOne`	Sets the system contact string.
Step 4	**snmp-server location** *text* **Example:** `Router(config)# snmp-server location LocationOne`	Sets the system location string.
Step 5	**snmp-server chassis-id** *number* **Example:** `Router(config)# snmp-server chassis-id 015A619T`	Sets the system serial number.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show snmp contact** **Example:** `Router# show snmp contact`	(Optional) Displays the contact strings configured for the system.
Step 8	**show snmp location** **Example:** `Router# show snmp location`	(Optional) Displays the location string configured for the system.
Step 9	**show snmp chassis** **Example:** `Router# show snmp chassis`	(Optional) Displays the system serial number.

Configuring SNMP Versions 1 and 2

When you configure SNMP versions 1 and 2, you can optionally create or modify views for community strings to limit which MIB objects an SNMP manager can access.

Perform the following tasks when configuring SNMP version 1 or version 2.

Prerequisites

- An established SNMP community string that defines the relationship between the SNMP manager and the agent.
- A host defined to be the recipient of SNMP notifications.

Creating or Modifying an SNMP View Record

You can assign views to community strings to limit which MIB objects an SNMP manager can access. You can use a predefined view or create your own view. If you are using a predefined view or no view at all, skip this task.

Perform this task to create or modify an SNMP view record.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server view** *view-name oid-tree* {**included** | **excluded**}
4. **no snmp-server view** *view-name oid-tree* {**included** | **excluded**}
5. **exit**
6. **show snmp view**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server view** *view-name oid-tree* {**included** | **excluded**} **Example:** `Router(config)# snmp-server view mib2 mib-2 included`	Creates a view record. • In this example, the mib2 view that includes all objects in the MIB-II subtree is created. **Note** You can use this command multiple times to create the same view record. If a view record for the same OID value is created multiple times, the latest entry of the object identifier takes precedence.
Step 4	**no snmp-server view** *view-name oid-tree* {**included** | **excluded**} **Example:** `Router(config)# no snmp-server view mib2 mib-2 included`	Removes a server view.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 6	**show snmp view** **Example:** `Router# show snmp view`	(Optional) Displays a view of the MIBs associated with SNMP.

Examples

The following example shows the SNMP view for the system.1.0 OID tree:

```
Router# show snmp view

test system.1.0 - included nonvolatile active
*ilmi system - included permanent active
*ilmi atmForumUni - included permanent active
v1default iso - included permanent active
v1default internet - included permanent active
v1default snmpUsmMIB - excluded permanent active
v1default snmpVacmMIB - excluded permanent active
v1default snmpCommunityMIB - excluded permanent active
v1default ciscoIpTapMIB - excluded permanent active
v1default ciscoMgmt.395 - excluded permanent active
v1default ciscoTap2MIB - excluded permanent active
```

Creating or Modifying Access Control for an SNMP Community

Use an SNMP community string to define the relationship between the SNMP manager and the agent. The community string acts like a password to regulate access to the agent on the router. Optionally, you can specify one or more of the following 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.

Perform this task to create or modify a community string.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*]
4. **no snmp-server community** *string*
5. **exit**
6. **show snmp community**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*] **Example:** `Router(config)# snmp-server community comaccess ro 4`	Defines the community access string. • You can configure one or more community strings.
Step 4	**no snmp-server community** *string* **Example:** `Router(config)# no snmp-server community comaccess`	Removes the community string from the configuration.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Command or Action	Purpose
Step 6 **show snmp community** **Example:** `Router# show snmp community`	(Optional) Displays the community access strings configured for the system.

Examples

The following example shows the community access strings configured to enable access to the SNMP manager:

```
Router# show snmp community

Community name: private
Community Index: private
Community SecurityName: private
storage-type: nonvolatile        active
Community name: private@1
Community Index: private@1
Community SecurityName: private
storage-type: read-only  active
Community name: public
Community Index: public
Community SecurityName: public
storage-type: nonvolatile        active
```

Configuring a Recipient of an SNMP Trap Operation

SNMP traps are unreliable because the receiver does not send acknowledgments when it receives traps. The sender does not know if the traps were received. However, an SNMP entity that receives an inform acknowledges the message with an SNMP response PDU. If the sender never receives the response, the inform can be sent again. Thus, informs are more likely to reach their intended destination.

Compared to traps, informs consume more resources in the agent and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform must be held in memory until a response is received or the request times out. Also, traps are sent only once; an inform may be sent several times. The retries increase traffic and overhead on the network.

If you do not enter a **snmp-server host** command, no notifications are sent. To configure the router to send SNMP notifications, you must enter at least one **snmp-server host** command. If you enter the command without keywords, all trap types are enabled for the host.

To enable multiple hosts, you must issue a separate **snmp-server host** command for each host. You can specify multiple notification types in the command for each host.

When multiple **snmp-server host** commands are given for the same host and type of notification, each succeeding command overwrites the previous command. Only the last **snmp-server host** command will be in effect. For example, if you enter an **snmp-server host inform** command for a host and then enter another **snmp-server host inform** command for the same host, the second command replaces the first.

The **snmp-server host** command is used in conjunction with the **snmp-server enable** command. Use the **snmp-server enable** command to specify which SNMP notifications are sent globally. For a host to receive most notifications, at least one **snmp-server enable** command and the **snmp-server host** command for that host must be enabled.

Some notification types cannot be controlled with the **snmp-server enable** command. For example, some notification types are always enabled and others are enabled by a different command. For example, the

linkUpDown notifications are controlled by the **snmp trap link-status** command. These notification types do not require an **snmp-server enable** command.

A *notification-type* option's availability depends on the router type and the Cisco IOS software features supported on the router. For example, the envmon notification type is available only if the environmental monitor is part of the system. To see what notification types are available on your system, use the command help (?) at the end of the **snmp-server host** command.

Perform this task to configure the recipient of an SNMP trap operation.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server host** *host-id* [**traps** | **informs**] [**version** {**1**| **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port-number*] [*notification-type*]
4. **exit**
5. **show snmp host**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server host** *host-id* [**traps** | **informs**] [**version** {**1**| **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port-number*] [*notification-type*] **Example:** `Router(config)# snmp-server host 172.16.1.27 version 2c public`	Specifies whether you want the SNMP notifications sent as traps or informs, the version of SNMP to use, the security level of the notifications (for SNMPv3), and the recipient (host) of the notifications.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

	Command or Action	Purpose
Step 5	**show snmp host** **Example:** `Router# show snmp host`	(Optional) Displays the SNMP notifications sent as traps, the version of SNMP, and the host IP address of the notifications.

Examples

The following example shows the host information configured for SNMP notifications:

```
Router> enable
Router# configure terminal
Router(config)# snmp-server host 10.2.28.1 inform version 2c public
Router(config)# exit
Router# show snmp host

Notification host: 10.2.28.1 udp-port: 162   type: inform
user: public    security model: v2c
traps: 00001000.00000000.00000000
```

Configuring SNMP Version 3

When you configure SNMPv3 and you want to use the SNMPv3 security mechanism for handling SNMP packets, you must establish SNMP groups and users with passwords.

Perform the following tasks to configure SNMPv3.

- Specifying SNMP-Server Group Names, page 44
- Configuring SNMP Server Users, page 46

Specifying SNMP-Server Group Names

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.

No default values exist for authentication or privacy algorithms when you configure the **snmp-server group** command. Also, no default passwords exist. For information about specifying a MD5 password, see the documentation for the **snmp-server user** command.

Perform this task to specify a new SNMP group or a table that maps SNMP users to SNMP views.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server group** [*groupname* {**v1** | **v2c** | **v3** [**auth** | **noauth** | **priv**]}] [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*]
4. **exit**
5. **show snmp group**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server group** [*groupname* {**v1** | **v2c** | **v3** [**auth** | **noauth** | **priv**]}] [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **Example:** `Router(config)# snmp-server group group1 v3 auth access lmnop`	Configures the SNMP server group to enable authentication for members of a specified named access list. • In this example, the SNMP server group *group1* is configured to enable user authentication for members of the named access list *lmnop*.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 5	**show snmp group** **Example:** `Router# show snmp group`	Displays information about each SNMP group on the network.

Examples

The following example shows information about each SNMP group on the network:

```
Router# show snmp group

groupname: V1                             security model:v1
readview : v1default                      writeview: <no writeview specified>
notifyview: <no notifyview specified>
row status: active
groupname: ILMI                           security model:v1
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active
groupname: ILMI                           security model:v2c
readview : *ilmi                          writeview: *ilmi
notifyview: <no notifyview specified>
row status: active
groupname: group1                         security model:v1
readview : v1default                      writeview: <no writeview specified>
notifyview: <no notifyview specified>
row status: active
```

Configuring SNMP Server Users

To configure a remote user, specify the IP address or port number for the remote SNMP agent of the device where the user resides. Also, before you configure remote users for a particular agent, configure the SNMP engine ID using the **snmp-server engineID** command with the remote option. The remote agent's SNMP engine ID is required when computing the authentication and privacy digests from the password. If the remote engine ID is not configured first, the configuration command will fail.

For the *privpassword* and *auth-password* arguments, the minimum length is one character; the recommended length is at least eight characters, and should include both letters and numbers.

SNMP passwords are localized using the SNMP engine ID of the authoritative SNMP engine. For informs, the authoritative SNMP agent is the remote agent. You must configure the remote agent's SNMP engine ID in the SNMP database before you can send proxy requests or informs to it.

Note Changing the engine ID after configuring the SNMP user does not allow the removal of the user. To remove the configurations, you need to first reconfigure all the SNMP configurations.

No default values exist for authentication or privacy algorithms when you configure the command. Also, no default passwords exist. The minimum length for a password is one character, although we recommend using at least eight characters for security. If you forget a password, you cannot recover it and will need to reconfigure the user. You can specify either a plain text password or a localized MD5 digest.

If you have the localized MD5 or SHA digest, you can specify that string instead of the plain text password. The digest should be formatted as aa:bb:cc:dd where aa, bb, and cc are hexadecimal values. Also, the digest should be exactly 16 octets in length.

Perform this task to add a new user to an SNMP group.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server engineID** {**local** *engine-id* | **remote** *ip-address* [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engine-id-string*}
4. **snmp-server user** *username groupname* [**remote** *ip-address* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*]
5. **exit**
6. **show snmp user** [*username*]
7. **show snmp engineID**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server engineID** {**local** *engine-id* | **remote** *ip-address* [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engine-id-string*} **Example:** `Router(config)# snmp-server engineID remote 172.12.15.4 udp-port 120 1a2833c0129a`	Configures the SNMP engine ID. • In this example, the SNMP engine ID is configured for a remote user.
Step 4	**snmp-server user** *username groupname* [**remote** *ip-address* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*] **Example:** `Router(config)# snmp-server user user1 group1 v3 auth md5 password123`	Configures a new user to an SNMP group with the plain text password "password123" for the user "user1" in the SNMPv3 group "group1".
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show snmp user** [*username*] **Example:** `Router# show snmp user user123`	Displays information about configured characteristics of an SNMP user.
Step 7	**show snmp engineID** **Example:** `Router# show snmp engineID`	(Optional) Displays information about the SNMP engine ID configured for an SNMP user.

Examples

The following example shows the SNMP engine ID configured for the remote user:

```
Router# show snmp engineID

Local SNMP engineID: 1A2836C0129A
Remote Engine ID          IP-addr    Port
1A2833C0129A           remote  10.2.28.1 120
```

The following example shows the information about the configured characteristics of the SNMP user1:

```
Router# show snmp user user1

User name: user1
```

```
Engine ID: 000000090200000000C025808
storage-type: nonvolatile       active access-list: 10
Rowstatus: active
Authentication Protocol: MD5
Privacy protocol: DES
Group name: group1
```

Configuring a Router as an SNMP Manager

The SNMP manager feature allows a router to act as a network management station--an SNMP client. As an SNMP manager, the router can send SNMP requests to agents and receive SNMP responses and notifications from agents. When the SNMP manager process is enabled, the router can query other SNMP agents and process incoming SNMP traps.

Most network security policies assume that routers will accept SNMP requests, send SNMP responses, and send SNMP notifications.

With the SNMP manager functionality enabled, the router may also send SNMP requests, receive SNMP responses, and receive SNMP notifications. Your security policy implementation may need to be updated prior to enabling this feature.

SNMP requests typically are sent to UDP port 161. SNMP responses are typically sent from UDP port 161. SNMP notifications are typically sent to UDP port 162.

Sessions are created when the SNMP manager in the router sends SNMP requests, such as informs, to a host or receives SNMP notifications from a host. One session is created for each destination host. If there is no further communication between the router and host within the session timeout period, the session will be deleted.

The router tracks statistics, such as the average round-trip time required to reach the host, for each session. Using the statistics for a session, the SNMP manager in the router can set reasonable timeout periods for future requests, such as informs, for that host. If the session is deleted, all statistics are lost. If another session with the same host is later created, the request timeout value for replies will return to the default value.

Sessions consume memory. A reasonable session timeout value should be large enough that regularly used sessions are not prematurely deleted, yet small enough such that irregularly used or one-time sessions are purged expeditiously.

Perform this task to enable the SNMP manager process and to set the session timeout value.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server manager**
4. **snmp-server manager session-timeout** *seconds*
5. **exit**
6. **show snmp**
7. **show snmp sessions** [**brief**]
8. **show snmp pending**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server manager** **Example:** `Router(config)# snmp-server manager`	Enables the SNMP manager.
Step 4	**snmp-server manager session-timeout** *seconds* **Example:** `Router(config)# snmp-server manager session-timeout 30`	(Optional) Changes the session timeout value.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 6	**show snmp** **Example:** `Router# show snmp`	(Optional) Displays the status of SNMP communications.
Step 7	**show snmp sessions** [**brief**] **Example:** `Router# show snmp sessions`	(Optional) Displays the status of SNMP sessions.
Step 8	**show snmp pending** **Example:** `Router# show snmp pending`	(Optional) Displays the current set of pending SNMP requests.

Examples

The following example shows the status of SNMP communications:

```
Router# show snmp

Chassis: 01506199
37 SNMP packets input
    0 Bad SNMP version errors
    4 Unknown community name
    0 Illegal operation for community name supplied
    0 Encoding errors
    24 Number of requested variables
    0 Number of altered variables
    0 Get-request PDUs
    28 Get-next PDUs
    0 Set-request PDUs
78 SNMP packets output
    0 Too big errors (Maximum packet size 1500)
    0 No such name errors
    0 Bad values errors
    0 General errors
    24 Response PDUs
    13 Trap PDUs
SNMP logging: enabled
    Logging to 172.17.58.33.162, 0/10, 13 sent, 0 dropped.
SNMP Manager-role output packets
    4 Get-request PDUs
    4 Get-next PDUs
    6 Get-bulk PDUs
    4 Set-request PDUs
    23 Inform-request PDUs
    30 Timeouts
    0 Drops
SNMP Manager-role input packets
    0 Inform response PDUs
    2 Trap PDUs
    7 Response PDUs
    1 Responses with errors
SNMP informs: enabled
    Informs in flight 0/25 (current/max)
    Logging to 172.17.217.141.162
        4 sent, 0 in-flight, 1 retries, 0 failed, 0 dropped
    Logging to 172.17.58.33.162
        0 sent, 0 in-flight, 0 retries, 0 failed, 0 dropped
```

The following example displays the status of SNMP sessions:

```
Router# show snmp sessions

Destination: 172.17.58.33.162, V2C community: public
  Round-trip-times: 0/0/0 (min/max/last)
  packets output
    0 Gets, 0 GetNexts, 0 GetBulks, 0 Sets, 4 Informs
    0 Timeouts, 0 Drops
  packets input
    0 Traps, 0 Informs, 0 Responses (0 errors)
Destination: 172.17.217.141.162, V2C community: public, Expires in 575 secs
  Round-trip-times: 1/1/1 (min/max/last)
  packets output
    0 Gets, 0 GetNexts, 0 GetBulks, 0 Sets, 4 Informs
    0 Timeouts, 0 Drops
 packets input
    0 Traps, 0 Informs, 4 Responses (0 errors)
```

The following example shows the current set of pending SNMP requests:

```
Router# show snmp pending

req id: 47, dest: 172.17.58.33.161, V2C community: public, Expires in 5 secs
req id: 49, dest: 172.17.58.33.161, V2C community: public, Expires in 6 secs
req id: 51, dest: 172.17.58.33.161, V2C community: public, Expires in 6 secs
req id: 53, dest: 172.17.58.33.161, V2C community: public, Expires in 8 secs
```

Enabling the SNMP Agent Shutdown Mechanism

Using SNMP packets, a network management tool can send messages to users on virtual terminals and on the console. This facility operates in a similar fashion to the **send** EXEC command; however, the SNMP request that causes the message to be issued to users also specifies the action to be taken after the message is delivered. One possible action is a shutdown request. After a system is shut down, typically, it is reloaded. Because the ability to cause a reload from the network is a powerful feature, it is protected by the **snmp-server system-shutdown** global configuration command. If you do not issue this command, the shutdown mechanism is not enabled.

Perform this task to enable the SNMP agent shutdown mechanism.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server system-shutdown**
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server system-shutdown** **Example:** `Router(config)# snmp-server system-shutdown`	Enables system shutdown using the SNMP message reload feature.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Defining the Maximum SNMP Agent Packet Size

You can define the maximum packet size permitted when the SNMP agent is receiving a request or generating a reply.

Perform this task to set the maximum permitted packet size.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server packetsize** *byte-count*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server packetsize** *byte-count* **Example:** `Router(config)# snmp-server packetsize 512`	Establishes the maximum packet size.
Step 4	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Limiting the Number of TFTP Servers Used via SNMP

You can limit the number of TFTP servers used for saving and loading configuration files via SNMP by using an access list. Limiting the use of TFTP servers in this way conserves system resources and centralizes the operation for manageability.

Perform this task to limit the number of TFTP servers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server tftp-server-list** *number*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server tftp-server-list** *number* **Example:** `Router(config)# snmp-server tftp-server-list 12`	Limits the number of TFTP servers used for configuration file copies via SNMP to the servers in an access list.

• Troubleshooting Tips, page 53

Troubleshooting Tips

To monitor SNMP trap activity in real time for the purposes of troubleshooting, use the SNMP **debug** commands, including the **debug snmp packet** EXEC command. For documentation of SNMP **debug** commands, see the *Cisco IOS Debug Command Reference*.

Disabling the SNMP Agent

Perform this task to disable any version of an SNMP agent.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **no snmp-server**

DETAILED STEPS

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

Configuring SNMP Notifications

To configure a router to send SNMP traps or informs, perform the tasks described in the following sections:

Note Many snmp-server commands use the word **traps** in their command syntax. Unless there is an option within the command to specify either traps or informs, the keyword **traps** should be taken to mean traps, informs, or both. Use the **snmp-server host** command to specify whether you want SNMP notifications to be sent as traps or informs. To use informs, the SNMP manager (also known as the SNMP proxy manager) must be available and enabled on a device. Earlier, the SNMP manager was available only with Cisco IOS PLUS images. However, the SNMP manager is now available with all Cisco IOS releases that support SNMP. Use Cisco Feature Navigator for information about SNMP manager support for Cisco IOS releases. To access Cisco Feature Navigator, go to http://www.cisco.com/go/cfn.

Configuring the Router to Send SNMP Notifications

Perform this task to configure the router to send traps or informs to a host.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server engineID remote** *remote-ip-address remote-engineID*
4. **snmp-server user** *username groupname* [**remote** *host* [**udp-port** *port*] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*]
5. **snmp-server group** *groupname* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*]
6. **snmp-server host** *host* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [*notification-type*]
7. **snmp-server enable traps** [*notification-type* [*notification-options*]]
8. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server engineID remote** *remote-ip-address remote-engineID* **Example:** `Router(config)# snmp-server engineID remote 172.16.20.3 80000009030000B064EFE100`	Specifies the SNMP engine ID and configures the VRF name traps-vrf for SNMP communications with the remote device at 172.16.20.3.
Step 4	**snmp-server user** *username groupname* [**remote** *host* [**udp-port** *port*] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*] **Example:** `Router(config)# snmp-server user abcd public remote 172.16.20.3 v3 encrypted auth md5 publichost remotehostusers`	Configures an SNMP user to be associated with the host created in Step 3. **Note** You cannot configure a remote user for an address without first configuring the engine ID for that remote host. This restriction is imposed in the design of these commands; if you try to configure the user before the host, you will receive a warning message and the command will not be executed.

	Command or Action	Purpose
Step 5	**snmp-server group** *groupname* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **Example:** `Router(config)# snmp-server group GROUP1 v2c auth read viewA write viewA notify viewB`	Configures an SNMP group.
Step 6	**snmp-server host** *host* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [*notification-type*] **Example:** `Router(config)# snmp-server host example.com informs version 3 public`	Specifies whether you want the SNMP notifications sent as traps or informs, the version of SNMP to use, the security level of the notifications (for SNMPv3), and the recipient (host) of the notifications. • The **snmp-server host** command specifies which hosts will receive SNMP notifications, and whether you want the notifications sent as traps or informs.
Step 7	**snmp-server enable traps** [*notification-type* [*notification-options*]] **Example:** `Router(config)# snmp-server enable traps bgp`	Enables sending of traps or informs and specifies the type of notifications to be sent. • If a *notification-type* is not specified, all supported notifications will be enabled on the router. • To discover which notifications are available on your router, enter the **snmp-server enable traps ?** command. • The **snmp-server enable traps** command globally enables the production mechanism for the specified notification types (such as Border Gateway Protocol [BGP] traps, config traps, entity traps, Hot Standby Router Protocol [HSRP] traps, and so on).
Step 8	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

Changing Notification Operation Values

You can specify a value other than the default for the source interface, message (packet) queue length for each host, or retransmission interval.

Perform this task to change notification operation values as needed.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server trap-source** *interface*
4. **snmp-server queue-length** *length*
5. **snmp-server trap-timeout** *seconds*
6. **snmp-server informs** [**retries** *retries*] [**timeout** *seconds*] [**pending** *pending*]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server trap-source** *interface* **Example:** `Router(config)# snmp-server trap-source ethernet 2/1`	Sets the IP address for the Ethernet interface in slot2, port 1 as the source for all SNMP notifications.
Step 4	**snmp-server queue-length** *length* **Example:** `Router(config)# snmp-server queue-length 50`	Establishes the message queue length for each notification. • This example shows the queue length set to 50 entries.
Step 5	**snmp-server trap-timeout** *seconds* **Example:** `Router(config)# snmp-server trap-timeout 30`	Defines how often to resend notifications on the retransmission queue.
Step 6	**snmp-server informs** [**retries** *retries*] [**timeout** *seconds*] [**pending** *pending*] **Example:** `Router(config)# snmp-server informs retries 10 timeout 30 pending 100`	Configures inform-specific operation values. • This example sets the maximum number of times to resend an inform, the number of seconds to wait for an acknowledgment before resending, and the maximum number of informs waiting for acknowledgments at any one time.

Controlling Individual RFC 1157 SNMP Traps

Starting with Cisco IOS Release 12.1(3)T, you can globally enable or disable authenticationFailure, linkUp, linkDown, warmStart, and coldStart traps or informs individually. (These traps constitute the "generic traps" defined in RFC 1157.) Note that linkUp and linkDown notifications are enabled by default on specific interfaces but will not be sent unless they are enabled globally.

Perform this task to enable the authenticationFailure, linkUp, linkDown, warmStart, and coldStart notification types.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server enable traps snmp** [**authentication**] [**linkup**] [**linkdown**] [**warmstart**] [**coldstart**]
4. **interface** *type slot/port*
5. **no snmp-server link status**
6. **exit**
7. **exit**
8. **show snmp mib ifmibtraps**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server enable traps snmp** [**authentication**] [**linkup**] [**linkdown**] [**warmstart**] [**coldstart**] **Example:** `Router(config)# snmp-server enable traps snmp`	Enables RFC 1157 generic traps. • When used without any of the optional keywords, enables authenticationFailure, linkUp, linkDown, warmStart, and coldStart traps. • When used with keywords, enables only the trap types specified. For example, to globally enable only linkUp and linkDown SNMP traps or informs for all interfaces, use the **snmp-server enable traps snmp linkup linkdown** form of this command.

	Command or Action	Purpose
Step 4	**interface** *type slot/port* **Example:** `Router(config)# interface ethernet 0/0`	Enters interface configuration mode for a specific interface. **Note** To enable SNMP traps for individual interfaces such as Dialer, use the **snmp trap link-status permit duplicates** command in interface configuration mode. For example, to enter dialer interface configuration mode, enter the interface type as dialer.
Step 5	**no snmp-server link status** **Example:** `Router(config-if)# no snmp-server link status`	Disables the sending of linkUp and linkDown notifications for all generic interfaces. **Note** To disable SNMP traps for individual interfaces such as Dialer, use the **no snmp trap link-status permit duplicates** command in interface configuration mode.
Step 6	**exit** **Example:** `Router(config-if)# exit`	Exits interface configuration mode.
Step 7	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**show snmp mib ifmibtraps** **Example:** `Router# show snmp mib ifmib traps`	(Optional) Displays the status of linkup and linkdown traps for each of the interfaces configured for the system.

Examples

The following example shows the status of linkup and linkdown traps for all interfaces configured for the system:

```
Router# show snmp mib ifmib traps

ifDescr                 ifindex    TrapStatus
-------------------------------------------------
FastEthernet3/6                14     enabled
FastEthernet3/19               27     enabled
GigabitEthernet5/1             57     enabled
unrouted VLAN 1005             73     disabled
FastEthernet3/4                12     enabled
FastEthernet3/39               47     enabled
FastEthernet3/28               36     enabled
FastEthernet3/48               56     enabled
unrouted VLAN 1003             74     disabled
FastEthernet3/2                10     enabled
Tunnel0                        66     enabled
SPAN RP Interface              64     disabled
Tunnel10                       67     enabled
FastEthernet3/44               52     enabled
GigabitEthernet1/3             3      enabled
FastEthernet3/11               19     enabled
FastEthernet3/46               54     enabled
```

```
GigabitEthernet1/1              1      enabled
FastEthernet3/13                21     enabled
unrouted VLAN 1                 70     disabled
GigabitEthernet1/4              4      enabled
FastEthernet3/9                 17     enabled
FastEthernet3/16                24     enabled
FastEthernet3/43                51     enabled
```

Configuring SNMP Notification Log Options

Perform this task to configure SNMP notification log options. These options allow you to control the log size and timing values. The SNMP log can become very large and long if left unmodified.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib notification-log default**
4. **snmp mib notification-log globalageout** *seconds*
5. **snmp mib notification-log globalsize** *size*
6. **exit**
7. **show snmp mib notification-log**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib notification-log default** **Example:** `Router(config)# snmp mib notification-log default`	Creates an unnamed SNMP notification log.
Step 4	**snmp mib notification-log globalageout** *seconds* **Example:** `Router(config)# snmp mib notification-log globalageout 20`	Sets the maximum amount of time for which the SNMP notification log entries remain in the system memory. • In this example, the system is configured to delete entries in the SNMP notification log that were logged more than 20 minutes ago.

	Command or Action	Purpose
Step 5	**snmp mib notification-log globalsize** *size* **Example:** `Router(config)# snmp mib notification-log globalsize 600`	Sets the maximum number of entries that can be stored in all SNMP notification logs.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show snmp mib notification-log** **Example:** `Router# show snmp mib notification-log`	Displays information about the state of the local SNMP notification logging.

Examples

This example shows information about the state of local SNMP notification logging:

```
Router# show snmp mib notification-log

GlobalAgeout 20, GlobalEntryLimit 600
Total Notifications logged in all logs 0
Log Name"", Log entry Limit 600, Notifications logged 0
Logging status enabled
Created by cli
```

Configuring Interface Index Display and Interface Indexes and Long Name Support

The display of Interface Indexes lets advanced users of SNMP view information about the interface registrations directly on a managed agent. An external NMS is not required.

Configuration of Long Alias Names for the interfaces lets users configure the ifAlias (the object defined in the MIB whose length is restricted to 64) up to 255 bytes.

SNMP must be enabled on your system.

The Interface Index Display and Interface Alias Long Name Support feature is not supported on all Cisco platforms. Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support.

Perform this task to configure the IF-MIB to retain ifAlias values of longer than 64 characters and to configure the ifAlias values for an interface.

Note To verify if the ifAlias description is longer than 64 characters, perform an SNMP MIB walk for the ifMIB ifAlias variable from an NMS and verify that the entire description is displayed in the values for ifXEntry. 18.
The description for interfaces also appears in the output from the **more system:running config** privileged EXEC mode command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp ifmib ifalias long**
4. **interface** *type number*
5. **description** *text-string*
6. **exit**
7. **show snmp mib**
8. **show snmp mib ifmib ifindex** [*type number*] [**detail**] [**free-list**]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp ifmib ifalias long** **Example:** `Router(config)# snmp ifmib ifalias long`	Configures the Interfaces MIB (IF-MIB) on the system to return ifAlias values of longer than 64 characters to a Network Management System. • If the ifAlias values are not configured using the **snmp ifmib ifalias long** command, the ifAlias description will be restricted to 64 characters.
Step 4	**interface** *type number* **Example:** `Router(config)# interface ethernet 2/4`	Enters interface configuration mode. • The form of this command varies depending on the interface being configured.

	Command or Action	Purpose
Step 5	**description** *text-string* **Example:** `Router(config)# description This text string description can be up to 256 characters long`	Configures a free-text description of the specified interface. • This description can be up to 240 characters in length and is stored as the ifAlias object value in the IF-MIB. • If the ifAlias values are not configured using the **snmp ifmib ifalias long** command, the ifAlias description for SNMP set and get operations is restricted to 64 characters, although the interface description is configured for more than 64 characters by using the **description** command.
Step 6	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 7	**show snmp mib** **Example:** `Router# show snmp mib`	Displays a list of MIB module instance identifiers registered on your system. • The resulting display could be lengthy.
Step 8	**show snmp mib ifmib ifindex** [*type number*] [**detail**] [**free-list**] **Example:** `Router# show snmp mib ifmib ifindex Ethernet 2/0`	Displays the Interfaces MIB ifIndex values registered on your system for all interfaces or the specified interface.

Examples

The following example lists the MIB module instance identifiers registered on your system. The resulting display could be lengthy. Only a small portion is shown here.

```
Router# show snmp mib
system.1
system.2
sysUpTime
system.4
system.5
system.6
system.7
system.8
sysOREntry.2
sysOREntry.3
sysOREntry.4
interfaces.1
ifEntry.1
ifEntry.2
ifEntry.3
ifEntry.4
ifEntry.5
ifEntry.6
ifEntry.7
ifEntry.8
ifEntry.9
```

```
ifEntry.10
ifEntry.11
 --More--
captureBufferEntry.2
captureBufferEntry.3
captureBufferEntry.4
captureBufferEntry.5
captureBufferEntry.6
captureBufferEntry.7
capture.3.1.1
eventEntry.1
eventEntry.2
eventEntry.3
eventEntry.4
eventEntry.5
eventEntry.6
eventEntry.7
logEntry.1
logEntry.2
logEntry.3
logEntry.4
rmon.10.1.1.2
rmon.10.1.1.3
rmon.10.1.1.4
rmon.10.1.1.5
rmon.10.1.1.6
rmon.10.1.1.7
rmon.10.2.1.2
rmon.10.2.1.3
rmon.10.3.1.2
```

The following example shows output for the Interfaces MIB ifIndex values registered on a system for a specific interface:

```
Router# show snmp mib ifmib ifindex Ethernet 2/0
Ethernet2/0: Ifindex = 2
```

The following example shows output for the Interfaces MIB ifIndex values registered on a system for all interfaces:

```
Router# show snmp mib ifmib ifindex
ATM1/0: Ifindex = 1
ATM1/0-aal5 layer: Ifindex = 12
ATM1/0-atm layer: Ifindex = 10
ATM1/0.0-aal5 layer: Ifindex = 13
ATM1/0.0-atm subif: Ifindex = 11
ATM1/0.9-aal5 layer: Ifindex = 32
ATM1/0.9-atm subif: Ifindex = 31
ATM1/0.99-aal5 layer: Ifindex = 36
ATM1/0.99-atm subif: Ifindex = 35
Ethernet2/0: Ifindex = 2
Ethernet2/1: Ifindex = 3
Ethernet2/2: Ifindex = 4
Ethernet2/3: Ifindex = 5
Null0: Ifindex = 14
Serial3/0: Ifindex = 6
Serial3/1: Ifindex = 7
Serial3/2: Ifindex = 8
Serial3/3: Ifindex = 9
```

- Troubleshooting Tips, page 64

Troubleshooting Tips

An alternative to using the ifAlias value for the identification of interfaces across reboots is to use the cciDescr object in the Cisco Circuit Interface MIB (CISCO-CIRCUIT-INTERFACE-MIB.my). This MIB object can be used only for circuit-based interfaces such as ATM or Frame Relay interfaces. Cisco IOS Release 12.2(2)T introduced the Circuit Interface Identification Persistence for SNMP feature, which maintains the user-defined name of the circuit (defined in the cciDescr object) across reboots, allowing for the consistent identification of circuit-based interfaces.

Configuring SNMP Support for VPNs

This section describes how to configure SNMP support for VPNs. The SNMP Support for VPNs feature provides configuration commands that allow users to associate SNMP agents and managers with specific VRFs. The specified VRF is used to send SNMP traps and informs and responses between agents and managers. If a VRF is not specified, the default routing table for the VPN is used.

Support for VPNs allows users to configure an SNMP agent to only accept SNMP requests from a certain set of VPNs. With this configuration, providers can provide network management services to their customers who then can manage all user-VPN devices.

Note

- This feature is not supported on all Cisco platforms. Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support.
- Not all MIBs are VPN-aware. To list the VPN-aware MIBs, use the **show snmp mib context** command. For more information about VPN-aware MIBs, see the SNMP Support over VPNs--Context-based Access Control configuration module.

Perform this task to configure SNMP support for a specific VPN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server host** *host-address* [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1**| **2c**| **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*]
4. **snmp-server engineID remote** *ip-address* [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string*
5. **exit**
6. **show snmp host**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server host** *host-address* [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1**| **2c**| **3** [**auth** | **noauth** |**priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] **Example:** `Router(config)# snmp-server host example.com public vrf trap-vrf`	Specifies the recipient of an SNMP notification operation and specifies the VRF table to be used for sending SNMP notifications.

Step 4	**snmp-server engineID remote** *ip-address* [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string*
	
	Example:
	`Router(config)# snmp-server engineID remote 172.16.20.3 vrf traps-vrf`
	
	Example:
	`80000009030000B064EFE100`
		Configures a name for the remote SNMP engine on a router when configuring SNMP over a specific VPN for a remote SNMP user.

Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 6	**show snmp host** **Example:** `Router# show snmp host`	(Optional) Displays the SNMP configuration and verifies that the SNMP Support for VPNs feature is configured properly.

Configuring Interface Index Persistence

The following sections contain the tasks to configure Interface Index Persistence:

- Enabling and Disabling IfIndex Persistence Globally, page 66
- Enabling and Disabling IfIndex Persistence on Specific Interfaces, page 67

Enabling and Disabling IfIndex Persistence Globally

Perform this task to enable IfIndex persistence globally.

The configuration tasks described in this section assume that you have configured SNMP on your routing device and are using SNMP to monitor network activity using the Cisco IOS CLI and/or an NMS application.

The interface-specific ifIndex persistence command (**snmp ifindex persistence**) cannot be used on subinterfaces. A command applied to an interface is automatically applied to all subinterfaces associated with that interface.

Testing indicates that approximately 25 bytes of NVRAM storage are used by this feature per interface. There may be some boot delay exhibited on platforms with lower CPU speeds.

Note After ifIndex persistence commands have been entered, the configuration must be saved using the **copy running-config startup-config** EXEC mode command to ensure consistent ifIndex values.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server ifindex persist**
4. **no snmp-server ifindex persist**
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server ifindex persist** **Example:** `Router(config)# snmp-server ifindex persist`	Globally enables ifIndex values that will remain constant across reboots.
Step 4	**no snmp-server ifindex persist** **Example:** `Router(config)# no snmp-server ifindex persist`	Disables global ifIndex persistence.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Enabling and Disabling IfIndex Persistence on Specific Interfaces

Perform this task to configure ifIndex persistence only on a specific interface.

Tip Use the **snmp ifindex clear** command on a specific interface when you want that interface to use the global configuration setting for ifIndex persistence. This command clears any ifIndex configuration commands previously entered for that specific interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *type slot / port*
4. **snmp ifindex persist**
5. **no snmp ifindex persist**
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**interface** *type slot / port* **Example:** `Router(config)# interface ethernet 0/1`	Enters interface configuration mode for the specified interface. **Note** The syntax of the **interface** command will vary depending on the platform you are using.
Step 4	**snmp ifindex persist** **Example:** `Router(config-if)# snmp ifindex persist`	Enables an ifIndex value that is constant across reboots on the specified interface.

	Command or Action	Purpose
Step 5	**no snmp ifindex persist** **Example:** `Router(config-if)# no snmp ifindex persist`	Disables an ifIndex value that is constant across reboots on the specified interface.
Step 6	**end** **Example:** `Router(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Configuring MIB Persistence

Note Beginning with Cisco IOS Release 12.4(20)T, MIB persistence is automatic; manual configuration is not required.

The MIB Persistence features allow the SNMP data of a MIB to be persistent across reloads; that is, MIB information retains the same set of object values each time a networking device reboots. The following sections contain tasks for using Distributed Management Event and Expression MIB persistence.

Prerequisites

- SNMP is configured on your networking device.
- Values for Event MIB and Expression MIB have been configured.

Restrictions

- If the number of MIB objects to persist increases, the NVRAM storage capacity may be strained. Occasionally, the time taken to write MIB data to NVRAM may be longer than expected.
- The Distributed Management Event MIB Persistence feature is not supported on all Cisco platforms. Use Cisco Feature Navigator to find information about platform support and Cisco IOS software image support.

Enabling and Disabling Event MIB Persistence

Perform this task to configure Event MIB Persistence.

Note Event MIB Persistence is disabled by default.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib persist event**
4. **no snmp mib persist event**
5. **exit**
6. **write mib-data**
7. **copy running-config startup-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib persist event** **Example:** `Router(config)# snmp mib persist event`	Enables MIB Persistence for the Event MIB.
Step 4	**no snmp mib persist event** **Example:** `Router(config)# no snmp mib persist event`	(Optional) Disables MIB Persistence for the Event MIB.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

	Command or Action	Purpose
Step 6	**write mib-data** **Example:** `Router# write mib-data`	Saves the Event MIB Persistence configuration data to NVRAM.
Step 7	**copy running-config startup-config** **Example:** `Router# copy running-config startup-config`	Copies the running configuration to the startup configuration.

Enabling and Disabling Expression MIB Persistence

Perform this task to configure Expression MIB Persistence.

Note Expression MIB Persistence is disabled by default.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib persist expression**
4. **no snmp mib persist expression**
5. **exit**
6. **write mib-data**
7. **copy running-config startup-config**
8. **more system:running-config**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib persist expression** **Example:** `Router(config)# snmp mib persist expression`	Enables MIB Persistence for Expression MIB.
Step 4	**no snmp mib persist expression** **Example:** `Router(config)# no snmp mib persist expression`	(Optional) Disables MIB Persistence for Expression MIB.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.
Step 6	**write mib-data** **Example:** `Router# write mib-data`	Saves the Expression MIB Persistence configuration data to NVRAM.
Step 7	**copy running-config startup-config** **Example:** `Router# copy running-config startup-config`	Copies the running configuration to the startup configuration.
Step 8	**more system:running-config** **Example:** `Router# more system:running-config`	Displays the currently running configuration. • Use this command to verify the MIB persistence configuration.

Configuring Event MIB Using SNMP

The Event MIB can be configured using SNMP directly. In this procedure, the Event MIB is configured to monitor the delta values of ifInOctets for all interfaces once per minute. If any of the samples exceed the specified threshold, a trap notification will be sent.

There are no Cisco IOS software configuration tasks associated with the Event MIB. All configuration of Event MIB functionality must be performed though applications using SNMP. This section provides a sample configuration session using a network management application on an external device. See the Additional References section for information about configuring SNMP on your Cisco routing device.

All configuration of Event MIB functionality must be performed though applications using SNMP. The following section provides a step-by-step Event MIB configuration using SNMP research tools available for Sun workstations. The **setany** commands given below are executed using the SNMP application. Note that these commands are not Cisco IOS CLI commands. It is assumed that SNMP has been configured on your routing device.

In this configuration, the objective is to monitor ifInOctets for all interfaces. The Event MIB is configured to monitor the delta values of ifInOctets for all interfaces once per minute. If any of the samples exceed the specified threshold of 30, a Trap notification will be sent.

There are five parts to the following example:

Setting the Trigger in the Trigger Table

Perform this task to set the trigger in the trigger table.

SUMMARY STEPS

1. **setany -v2c $ADDRESS private mteTriggerEntryStatus.4.106.111.104.110.1 -i 5**
2. **setany -v2c $ADDRESS private mteTriggerValueID.4.106.111.104.110.1 -d 1.3.6.1.2.1.2.2.1.10**
3. **setany -v2c $ADDRESS private mteTriggerValueIDWildcard.4.106.111.104.110.1 -i 1**
4. **setany -v2c $ADDRESS private mteTriggerTest.4.106.111.104.110.1 -o '20'**
5. **setany -v2c $ADDRESS private mteTriggerFrequency.4.106.111.104.110.1 -g 60**
6. **setany -v2c $ADDRESS private mteTriggerSampleType.4.106.111.104.110.1 -i 2**
7. **setany -v2c $ADDRESS private mteTriggerEnabled.4.106.111.104.110.1 -i 1**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**setany -v2c $ADDRESS private mteTriggerEntryStatus.4.106.111.104.110.1 -i 5**	Creates a trigger row in the table with john as the mteOwner and 1 as the trigger name. • The index is given in decimal representation of the ASCII value of john.1.
Step 2	**setany -v2c $ADDRESS private mteTriggerValueID. 4.106.111.104.110.1 -d 1.3.6.1.2.1.2.2.1.10**	Sets the mteTriggerValueID to the OID to be watched. • In this example, the OID to be monitored is ifInOctets.
Step 3	**setany -v2c $ADDRESS private mteTriggerValueIDWildcard.4.106.111.104.110.1 -i 1**	Sets the mteTriggerValueIDWildcard to TRUE to denote a object referenced through wildcarding.
Step 4	**setany -v2c $ADDRESS private mteTriggerTest. 4.106.111.104.110.1 -o '20'**	Sets the mteTriggerTest to Threshold.
Step 5	**setany -v2c $ADDRESS private mteTriggerFrequency.4.106.111.104.110.1 -g 60**	Sets the mteTriggerFrequency to 60. This means that ifInOctets are monitored once every 60 seconds.
Step 6	**setany -v2c $ADDRESS private mteTriggerSampleType.4.106.111.104.110.1 -i 2**	Sets the sample type to Delta.
Step 7	**setany -v2c $ADDRESS private mteTriggerEnabled. 4.106.111.104.110.1 -i 1**	Enables the trigger.

Creating an Event in the Event Table

Perform this task to create an event in the event table.

SUMMARY STEPS

1. **setany -v2c $ADDRESS private mteEventEntryStatus.4.106.111.104.110.101.118.101.110. 116 -i 5**
2. **setany -v2c $ADDRESS private mteEventEnabled.4.106.111.104.110.101.118.101.110.116 -i 1**
3. **setany -v2c $ADDRESS private mteEventEntryStatus.4.106.111.104.110.101.118.101.110. 116 -i 1**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**setany -v2c $ADDRESS private mteEventEntryStatus. 4.106.111.104.110.101.118.101.110. 116 -i 5**	Creates a row in the Event Table. • The mteOwner here is again john, and the event is mteEventName. • The default action is to send out a notification.
Step 2	**setany -v2c $ADDRESS private mteEventEnabled. 4.106.111.104.110.101.118.101.110.116 -i 1**	Enables the Event.

	Command or Action	Purpose
Step 3	**setany -v2c $ADDRESS private mteEventEntryStatus. 4.106.111.104.110.101.118.101.110. 116 -i 1**	Makes the EventRow active.

Setting the Trigger Threshold in the Trigger Table

Perform this task to set the trigger threshold in the trigger table.

SUMMARY STEPS

1. **setany -v2c $ADDRESS private mteTriggerThresholdRising.4.106.111.104.110.1 -i 30**
2. **setany -v2c $ADDRESS private mteTriggerThresholdRisingEventOwner.4.106.111.104.110.1 -D "owner"**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**setany -v2c $ADDRESS private mteTriggerThresholdRising. 4.106.111.104.110.1 -i 30**	Sets the Rising Threshold value to 30. Note that a row would already exist for john.1 in the Trigger Threshold Table.
Step 2	**setany -v2c $ADDRESS private mteTriggerThresholdRisingEventOwner.4.106.111.104.110.1 -D "owner"** **Example:** `setany -v2c $ADDRESS private mteTriggerThresholdRisingEvent. 4.106.111.104.110.1 -D "event"`	Points to the entry in the Event Table that specifies the action to be performed.

Activating the Trigger

Perform this task to activate the trigger.

SUMMARY STEPS

1. **setany -v2c $ADDRESS private mteTriggerEntryStatus.4.106.111.104.110.1 -i 1**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**setany -v2c $ADDRESS private mteTriggerEntryStatus.4.106.111.104.110.1 -i 1**	Makes the trigger active.

To confirm that the above configuration is working, ensure that at least one of the interfaces gets more than 30 packets in a minute. This should cause a trap to be sent out after one minute.

Monitoring and Maintaining Event MIB

Use the following commands to monitor Event MIB activity from the Cisco IOS CLI:

Command	Purpose
debug management event mib	Prints messages to the screen whenever the Event MIB evaluates a specified trigger. These messages are given in realtime and are intended to be used by technical support engineers for troubleshooting purposes.
show management event	Displays the SNMP Event values that have been configured on your routing device through the use of the Event MIB.

Configuring Event MIB Using the CLI

The Event MIB can be configured using SNMP directly. In this procedure, the Event MIB is configured to monitor delta values of ifInOctets for all interfaces once per minute. If any of the samples exceed the specified threshold, a trap notification will be sent.

However, in Cisco IOS Release 12.4(20)T, the Event MIB feature is enhanced to add CLIs to configure the events, event action, and trigger.

This section contains the following tasks to configure the Event MIB:

Configuring Scalar Variables

Perform this task to configure scalar variables for the Event MIB.

To configure scalar variables for the Event MIB, you should be familiar with the Event MIB scalar variables.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib event sample minimum** *value*
4. **snmp mib event sample instance maximum** *value*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib event sample minimum** *value* **Example:** `Router(config)# snmp mib event sample minimum 10`	Sets the minimum value for object sampling.
Step 4	**snmp mib event sample instance maximum** *value* **Example:** `Router(config)# snmp mib event sample instance maximum 50`	Sets the maximum value for object instance sampling.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Configuring Event MIB Object List

To configure the Event MIB, you need to set up a list of objects that can be added to notifications according to the trigger, trigger test, or event.

To configure the Event MIB object list, you should be familiar with the Event MIB objects and object identifiers, which can be added to notifications according to the event, trigger, or trigger test.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib event object list owner** *object-list-owner* **name** *object-list-name object-number*
4. **object id** *object-identifier*
5. **wildcard**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib event object list owner** *object-list-owner* **name** *object-list-name object-number* **Example:** `Router(config)# snmp mib event object list owner owner1 name objectA number 10`	Configures the Event MIB object list.
Step 4	**object id** *object-identifier* **Example:** `Router(config-event-objlist)# object id ifInOctets`	Specifies the object identifier for the object configured for the event.
Step 5	**wildcard** **Example:** `Router(config-event-objlist)# wildcard`	(Optional) Starts a wildcarded search for object identifiers. By specifying a partial object identifier, you can obtain a list of object identifiers.
Step 6	**exit** **Example:** `Router(config-event-objlist)# exit`	Exits object list configuration mode.

Configuring Event

Perform this task to configure a management event.

To configure a management event, you should be familiar with SNMP MIB events and object identifiers.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib event owner** *event-owner* **name** *event-name*
4. **description** *event-description*
5. **enable**
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib event owner** *event-owner* **name** *event-name* **Example:** `Router(config)# snmp mib event owner owner1 event EventA`	Enters event configuration mode.
Step 4	**description** *event-description* **Example:** `Router(config-event)# description "EventA is an RMON event"`	Describes the function and use of the event.
Step 5	**enable** **Example:** `Router(config-event)# enable`	Enables the event. **Note** The event can be executed during an event trigger only if it is enabled.
Step 6	**exit** **Example:** `Router(config-event)# exit`	Exits event configuration mode.

Configuring Event Action

By configuring an event action, you can define the actions that an application can perform during an event trigger. The actions for an event include sending a notification, setting a MIB object and so on. You can set the event action information to either **set** or **notification**. The actions for the event can be configured only in event configuration mode.

The following sections contain the tasks to configure an event action:

- Configuring Action Notification, page 80
- Configuring Action Set, page 81

Configuring Action Notification

Perform this task to set the notification action for the event.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib event owner** *event-owner* **name** *event-name*
4. **action notification**
5. **object id** *object-id*
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib event owner** *event-owner* **name** *event-name* **Example:** `Router(config)# snmp mib event owner owner1 name test`	Enters event configuration mode.
Step 4	**action notification** **Example:** `Router(config-event)# action notification`	Sets the notification action for an event and enters action notification configuration mode. **Note** If the event action is set to notification, a notification is generated whenever an object associated with an event is modified.

	Command or Action	Purpose
Step 5	**object id** *object-id* **Example:** `Router(config-event-action-notification)# object id ifInOctets`	Configures an object for action notification. When the object specified is modified, a notification will be sent to the host system.
Step 6	**exit** **Example:** `Router(config-event-action-notification)# exit`	Exits action notification configuration mode.

Configuring Action Set

Perform this task to set actions for an event.

SUMMARY STEPS

1. **action set**
2. **object id** *object-id*
3. **value** *integer-value*
4. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**action set** **Example:** `Router(config-event)# action set`	Enters action set configuration mode.
Step 2	**object id** *object-id* **Example:** `Router(config-event-action-set)# object id ifInOctets`	Configures an object for action set. • When the object specified is modified, a specified action will be performed.
Step 3	**value** *integer-value* **Example:** `Router(config-event-action-set)# value 10`	Sets a value for the object.

	Command or Action	Purpose
Step 4	**exit** **Example:** `Router(config-event-action-set)# exit`	Exits action set configuration mode.

Configuring Event Trigger

By configuring an event trigger, you can list the objects to be monitored, and associate each trigger to an event.

Perform this task to configure an event trigger.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib event trigger owner** *trigger-owner* **name** *trigger-name*
4. **description** *trigger-description*
5. **frequency** *seconds*
6. **object list owner** *object-list-owner* **name** *object-list-name*
7. **object id** *object-identifier*
8. **enable**
9. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib event trigger owner** *trigger-owner* **name** *trigger-name* **Example:** `Router(config)# snmp mib event trigger owner owner1 name EventTriggerA`	Enables event trigger configuration mode for the specified event trigger.

	Command or Action	Purpose
Step 4	**description** *trigger-description* **Example:** `Router(config-event-trigger)# description EventTriggerA is an RMON alarm.`	Describes the function and use of the event trigger.
Step 5	**frequency** *seconds* **Example:** `Router(config-event-trigger)# frequency 120`	Configures the waiting time (number of seconds) between trigger samples.
Step 6	**object list owner** *object-list-owner* **name** *object-list-name* **Example:** `Router(config-event-trigger)# object list owner owner1 name ObjectListA`	Specifies the list of objects that can be added to notifications.
Step 7	**object id** *object-identifier* **Example:** `Router(config-event-trigger)# object id ifInOctets`	Configures object identifiers for an event trigger.
Step 8	**enable** **Example:** `Router(config-event-trigger)# enable`	Enables the event trigger.
Step 9	**exit** **Example:** `Router(config-event-trigger)# exit`	Exits event trigger configuration mode.

Configuring Existence Trigger Test

You should configure this trigger type in event trigger configuration mode.

Perform this task to configure trigger parameters for the test existence trigger type.

SUMMARY STEPS

1. **test existence**
2. **event owner** *event-owner* **name** *event-name*
3. **object list owner** *object-list-owner* **name** *object-list-name*
4. **type** {**present** | **absent** | **changed**}
5. **startup** {**present** | **absent**}
6. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**test existence** **Example:** `Router(config-event-trigger)# test existence`	Enables test existence configuration mode.
Step 2	**event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-existence)# event owner owner1 name EventA`	Configures the event for the existence trigger test.
Step 3	**object list owner** *object-list-owner* **name** *object-list-name* **Example:** `Router(config-event-trigger-existence)# object list owner owner1 name ObjectListA`	Configures the list of objects for the existence trigger test.
Step 4	**type** {**present** | **absent** | **changed**} **Example:** `Router(config-event-trigger-existence)# type present`	Performs the specified type of existence test. Existence tests are of the following three types: • Present--Setting type to present tests if the objects that appear during the event trigger exist. • Absent--Setting type to absent tests if the objects that disappear during the event trigger exist. • Changed--Setting type to changed tests if the objects that changed during the event trigger exist.
Step 5	**startup** {**present** | **absent**} **Example:** `Router(config-event-trigger-existence)# startup present`	Triggers an event if the test is performed successfully.

	Command or Action	Purpose
Step 6	**exit** **Example:** `Router(config-event-trigger-existence)# exit`	Exits existence trigger test configuration mode.

Configuring Boolean Trigger Test

You should configure this trigger test in event trigger configuration mode.

Perform this task to configure trigger parameters for the Boolean trigger type.

SUMMARY STEPS

1. **test boolean**
2. **comparison** {**unequal** | **equal** | **less** | **lessOrEqual** | **greater** | **greaterOrEqual**}
3. **value** *integer-value*
4. **object list owner** *object-list-owner* **name** *object-list-name*
5. **event owner** *event-owner* **name** *event-name*
6. **startup**
7. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**test boolean** **Example:** `Router(config-event-trigger)# test boolean`	Enables Boolean trigger test configuration mode.
Step 2	**comparison** {**unequal** | **equal** | **less** | **lessOrEqual** | **greater** | **greaterOrEqual**} **Example:** `Router(config-event-trigger-boolean)# comparison unequal`	Performs the specified Boolean comparison test. • The value for the Boolean comparison test can be set to unequal, equal, less, lessOrEqual, greater, or greaterOrEqual.
Step 3	**value** *integer-value* **Example:** `Router(config-event-trigger-boolean)# value 10`	Sets a value for the Boolean trigger test.

	Command or Action	Purpose
Step 4	**object list owner** *object-list-owner* **name** *object-list-name* **Example:** `Router(config-event-trigger-boolean)# object list owner owner1 name ObjectListA`	Configures the list of objects for the Boolean trigger test.
Step 5	**event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-boolean)# event owner owner1 name EventA`	Configures the event for the Boolean trigger type.
Step 6	**startup** **Example:** `Router(config-event-trigger-boolean)# startup`	Triggers an event if the test is performed successfully.
Step 7	**exit** **Example:** `Router(config-event-trigger-boolean)# exit`	Exits Boolean trigger test configuration mode.

Configuring Threshold Trigger Test

You should configure this trigger test in event trigger configuration mode.

Perform this task to configure trigger parameters for the threshold trigger test.

SUMMARY STEPS

1. **test threshold**
2. **object list owner** *object-list-owner* **name** *object-list-name*
3. **rising** *integer-value*
4. **rising event owner** *event-owner* **name** *event-name*
5. **falling** *integer-value*
6. **falling event owner** *event-owner* **name** *event-name*
7. **delta rising** *integer-value*
8. **delta rising event owner** *event-owner* **name** *event-name*
9. **delta falling** *integer-value*
10. **delta falling event owner** *event-owner* **name** *event-name*
11. **startup** {**rising** | **falling** | **rising-or-falling**}
12. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**test threshold** **Example:** `Router(config-event-trigger)# test threshold`	Enables threshold trigger test configuration mode.
Step 2	**object list owner** *object-list-owner* **name** *object-list-name* **Example:** `Router(config-event-trigger-threshold)# object list owner owner1 name ObjectListA`	Configures the list of objects for the threshold trigger test.
Step 3	**rising** *integer-value* **Example:** `Router(config-event-trigger-threshold)# rising 100`	Sets the rising threshold to the specified value.
Step 4	**rising event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-threshold)# rising event owner owner1 name EventA`	Configures an event for the threshold trigger test for the rising threshold.
Step 5	**falling** *integer-value* **Example:** `Router(config-event-trigger-threshold)# falling 50`	Sets the falling threshold to the specified value.
Step 6	**falling event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-threshold)# falling event owner owner1 name EventB`	Configures an event for the threshold trigger test for the falling threshold.
Step 7	**delta rising** *integer-value* **Example:** `Router(config-event-trigger-threshold)# delta rising 30`	Sets the delta rising threshold to the specified value when the sampling method specified for the event trigger is delta.

	Command or Action	Purpose
Step 8	**delta rising event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-threshold)# delta rising event owner owner1 name EventC`	Configures an event for the threshold trigger test for the delta rising threshold.
Step 9	**delta falling** *integer-value* **Example:** `Router(config-event-trigger-threshold)# delta falling 10`	Sets the delta falling threshold to the specified value when the sampling method specified for the event trigger is delta.
Step 10	**delta falling event owner** *event-owner* **name** *event-name* **Example:** `Router(config-event-trigger-threshold)# delta falling event owner owner1 name EventAA`	Configures an event for the threshold target test for the delta falling threshold.
Step 11	**startup** {**rising** | **falling** | **rising-or-falling**} **Example:** `Router(config-event-trigger-threshold)# startup rising`	Triggers an event when the threshold trigger test conditions are met.
Step 12	**exit** **Example:** `Router(config-event-trigger-threshold)# exit`	Exits threshold trigger test configuration mode.

Configuring Expression MIB Using SNMP

Expression MIB can be configured using SNMP directly.

There are no Cisco IOS software configuration tasks associated with Expression MIB. All configurations of the Expression MIB functionality must be performed though applications using SNMP. This section provides a sample configuration session using a network management application on an external device. See the Additional References section for information about configuring SNMP on your Cisco routing device.

The following section provides a step-by-step Expression MIB configuration using SNMP research tools available for Sun workstations. The **setany** commands given below are executed using the SNMP application. Note that these commands are not Cisco IOS CLI commands. It is assumed that SNMP has been configured on your routing device.

In the following configuration, a wildcarded expression involving the addition of the counters ifInOctects and ifOutOctects are evaluated.

SUMMARY STEPS

1. **setany -v2c $SNMP_HOST private expResourceDeltaMinimum.0 -i 60**
2. **setany -v2c $SNMP_HOST private expExpressionIndex.116.101.115.116 -g 9**
3. **setany -v2c $SNMP_HOST private expNameStatus.116.101.115.116 -i 5**
4. **setany -v2c $SNMP_HOST private expExpressionComment.9 -D "test expression"**
5. **setany -v2c $SNMP_HOST private expExpression.9 -D '$1 + $2'**
6. **setany -v2c $SNMP_HOST private expObjectID.9.1 -d ifInOctets**
7. **setany -v2c $SNMP_HOST private expObjectSampleType.9.1 -i 2**
8. **setany -v2c $SNMP_HOST private expObjectIDWildcard.9.1 -i 1**
9. **setany -v2c $SNMP_HOST private expObjectStatus.9.1 -i 1**
10. **setany -v2c $SNMP_HOST private expNameStatus.116.101.115.116 -i 1**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**setany -v2c $SNMP_HOST private expResourceDeltaMinimum.0 -i 60**	Sets the minimum delta interval that the system will accept.
Step 2	**setany -v2c $SNMP_HOST private expExpressionIndex.116.101.115.116 -g 9**	Sets the identification number used for identifying the expression. • For example, expName can be 'test', which is ASCII 116.101.115.116.
Step 3	**setany -v2c $SNMP_HOST private expNameStatus.116.101.115.116 -i 5**	Creates an entry in the expNameStatusTable. **Note** When an entry is created in the expNameTable, it automatically creates an entry in the expExpressionTable.
Step 4	**setany -v2c $SNMP_HOST private expExpressionComment.9 -D "test expression"**	Sets the object to a comment to explain the use or meaning of the expression. • Here, the comment is "test expression".
Step 5	**setany -v2c $SNMP_HOST private expExpression.9 -D '$1 + $2'**	Sets the object expExpression to an expression that needs to be evaluated. • In this expression, "$1" corresponds to "ifInOctets", "$2" corresponds to "ifOutOctets", and the expression signifies the addition of the two counter objects.
Step 6	**setany -v2c $SNMP_HOST private expObjectID.9.1 -d ifInOctets** **Example:** `setany -v2c $SNMP_HOST private expObjectID.9.2 -d ifOutOctets`	Specifies the object identifiers used in the expression mentioned in the above set for calculation. • Here, the number "9", suffixed to the object expObjectID, corresponds to the unique identifier used for identifying the expression, and the number "1" following "9" is another unique identifier used for identifying an object within the expression. Set the expObjectID to the two objects used in forming the expression.

	Command or Action	Purpose
Step 7	**setany -v2c $SNMP_HOST private expObjectSampleType.9.1 -i 2** **Example:** `setany -v2c $SNMP_HOST private expObjectSampleType.9.2 -i 2`	Sets the type of sampling to be done for objects in the expression. • There are two types of sampling: a) Absolute b) Delta. Here, the sample type has been set to "Delta".
Step 8	**setany -v2c $SNMP_HOST private expObjectIDWildcard.9.1 -i 1** **Example:** `setany -v2c $SNMP_HOST private expObjectIDWildcard.9.2 -i 1`	Specifies whether the expObjectID is wildcarded or not. In this case, both the expObjectID are wildcarded.
Step 9	**setany -v2c $SNMP_HOST private expObjectStatus.9.1 -i 1** **Example:** `setany -v2c $SNMP_HOST private expObjectStatus.9.2 -i 1`	Sets the rows in the expObjectTable to active.
Step 10	**setany -v2c $SNMP_HOST private expNameStatus.116.101.115.116 -i 1**	Sets the rows in the expNameTable to active so that the value of the expression can be evaluated. • The value of the expression can now be obtained from the expValueTable.

Configuring Expression MIB Using the CLI

Expression MIB can be configured using SNMP directly. However, in Cisco IOS Release 12.4(20)T, the Expression MIB feature is enhanced to add CLIs to configure expressions. You should be familiar with expressions, object identifiers, and sampling methods before configuring Expression MIB.

The following sections contain the tasks to configure Expression MIB:

- Configuring Expression MIB Scalar Objects, page 90
- Configuring Expressions, page 91

Configuring Expression MIB Scalar Objects

Expression MIB has the following scalar objects:

- expResourceDeltaMinimum
- expResourceDeltaWildcardInstanceMaximum

Perform this task to configure Expression MIB scalar objects.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib expression delta minimum** *seconds*
4. **snmp mib expression delta wildcard maximum** *number-of-instances*
5. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib expression delta minimum** *seconds* **Example:** `Router(config)# snmp mib expression delta minimum 20`	(Optional) Sets the minimum delta interval in seconds. **Note** Application may use larger values for this minimum delta interval to lower the impact of constantly computing deltas. For larger delta sampling intervals, the application samples less often and has less overhead. By using this command, you can enforce a lower overhead for all expressions created after the delta interval is set.
Step 4	**snmp mib expression delta wildcard maximum** *number-of-instances* **Example:** `Router(config)# snmp mib expression delta maximum 120`	(Optional) Limits the maximum number of dynamic instance entries for wildcarded delta objects in expressions. • For a given delta expression, the number of dynamic instances is the number of values that meet all criteria to exist, times the number of delta values in the expression. • There is no preset limit for the instance entries and it is dynamic based on a system's resources.
Step 5	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode.

Configuring Expressions

Perform this task to configure an expression.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp mib expression owner** *expression-owner* **name** *expression-name*
4. **description** *expression-description*
5. **expression** *expression*
6. **delta interval** *seconds*
7. **value type** {**counter32** | **unsigned32** | **timeticks** | **integer32** | **ipaddress** | **octetstring** | **objectid** | **counter64**}
8. **enable**
9. **object** *object-number*
10. **id** *object-identifier*
11. **wildcard**
12. **discontinuity object** *discontinuity-object-id* [**wildcard**] [**type** {**timeticks** | **timestamp** | **date-and-time**}]
13. **conditional object** *conditional-object-id* **[wildcard]**
14. **sample** {**absolute** | **delta** | **changed**}
15. **exit**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp mib expression owner** *expression-owner* **name** *expression-name* **Example:** `Router(config-expression)# snmp mib expression owner owner1 name ExpA`	Enables the expression to be configured.
Step 4	**description** *expression-description* **Example:** `Router(config-expression)# description this expression is created for the sysLocation MIB object`	Configures a description for the expression.

	Command or Action	Purpose
Step 5	**expression** *expression* **Example:** `Router(config-expression)# expression ($1+$2)*800/$3`	Configures the expression to be evaluated. **Note** The expressions are in ANSI C syntax. However, the variables in an expression are defined as a combination of the dollar sign ($) and an integer that corresponds to the object number of the object used in evaluating the expression.
Step 6	**delta interval** *seconds* **Example:** `Router(config-expression)# delta interval 180`	Configures the sampling interval for objects in the expression if the sampling method is delta.
Step 7	**value type** {**counter32** | **unsigned32** | **timeticks** | **integer32** | **ipaddress** | **octetstring** | **objectid** | **counter64**} **Example:** `Router(config-expression)# value type counter32`	Sets the specified value type for the expression.
Step 8	**enable** **Example:** `Router(config-expression)# enable`	Enables an expression for evaluation.
Step 9	**object** *object-number* **Example:** `Router(config-expression)# object 2`	Configures the objects that are used for evaluating an expression. • The object number is used to associate the object with the variables in the expression. The variable corresponding to the object is $ and object number. Thus, the variable in the example used here corresponds to $10.
Step 10	**id** *object-identifier* **Example:** `Router(config-expression-object)# id ifInOctets`	Configures the object identifier.
Step 11	**wildcard** **Example:** `Router(config-expression-object)# wildcard`	(Optional) Enables a wildcarded search for objects used in evaluating an expression.

	Command or Action	Purpose
Step 12	**discontinuity object** *discontinuity-object-id* [**wildcard**] [**type** {**timeticks** | **timestamp** | **date-and-time**}] **Example:** `Router(config-expression-object)# discontinuity object sysUpTime`	(Optional) Configures the discontinuity properties for the object if the object sampling type is set to delta or changed. The discontinuity object ID supports normal checking for a discontinuity in a counter. • Using the **wildcard** keyword, you can enable wildcarded search for objects with discontinuity properties. • Using the **type** keyword, you can set value for objects with discontinuity properties.
Step 13	**conditional object** *conditional-object-id* **[wildcard]** **Example:** `Router(config-expression-object)# conditional object mib-2.90.1.3.1.1.2.3.112.99.110.4.101.120.112.53`	(Optional) Configures the conditional object identifier. • Using the **wildcard** keyword, you can enable a wildcarded search for conditional objects with discontinuity properties.
Step 14	**sample** {**absolute** | **delta** | **changed**} **Example:** `Router(config-expression-object)# sample delta`	Enables the specified sampling method for the object. This example uses the delta sampling method. You can set any of the three sampling methods: absolute, delta, and changed. • Absolute sampling--Uses the value of the MIB object during sampling. • Delta sampling--Uses the last sampling value maintained in the application. This method requires applications to do continuous sampling. • Changed sampling--Uses the changed value of the object since the last sample.
Step 15	**exit** **Example:** `Router(config-expression-object)# exit`	Exits expression object configuration mode.

Configuration Examples for SNMP Support

Example Configuring SNMPv1 SNMPv2c and SNMPv3

The following example shows how to enable SNMPv1, SNMPv2c, and SNMPv3. The configuration permits any SNMP manager to access all objects with read-only permissions using the community string named public. This configuration does not cause the router to send traps.

```
snmp-server community public
```

The following example shows how to permit SNMP access to all objects with read-only permission using the community string named public. The router will also send ISDN traps to the hosts 172.16.1.111 and 172.16.1.33 using SNMPv1 and to the host 172.16.1.27 using SNMPv2c. The community string named public is sent with the traps.

```
snmp-server community public
snmp-server enable traps isdn
snmp-server host 172.16.1.27 version 2c public
snmp-server host 172.16.1.111 version 1 public
snmp-server host 172.16.1.33 public
```

The following example shows how to allow read-only access for all objects to members of access list 4 that specify the comaccess community string. No other SNMP managers have access to any objects. SNMP Authentication Failure traps are sent by SNMPv2c to the host example.com using the community string named public.

```
snmp-server community comaccess ro 4
snmp-server enable traps snmp authentication
snmp-server host example.com version 2c public
```

The following example shows how to configure a remote user to receive traps at the noAuthNoPriv security level when the SNMPv3 security model is enabled:

```
snmp-server group group1 v3 noauth
snmp-server user remoteuser1 group1 remote 10.12.8.4
snmp-server host 10.12.8.4 informs version 3 noauth remoteuser config
```

The following example shows how to configure a remote user to receive traps at the authNoPriv security level when the SNMPv3 security model is enabled:

```
snmp-server group group2 v3 auth
snmp-server user AuthUser group2 remote 10.12.8.4 v3 auth md5 password1
```

The following example shows how to configure a remote user to receive traps at the priv security level when the SNMPv3 security model is enabled:

```
snmp-server group group3 v3 priv
snmp-server user PrivateUser group3 remote 10.12.8.4 v3 auth md5 password1 priv access
des56
```

The following example shows how to send Entity MIB inform notifications to the host example.com. The community string is restricted. The first line enables the router to send Entity MIB notifications in addition to any traps or informs previously enabled. The second line specifies that the notifications should be sent as informs, specifies the destination of these informs, and overwrites the previous **snmp-server host** commands for the host example.com.

```
snmp-server enable traps entity
snmp-server host informs example.com restricted entity
```

The following example shows how to send SNMP and Cisco environmental monitor enterprise-specific traps to the address 172.30.2.160:

```
snmp-server enable traps
snmp-server host 172.30.2.160 public snmp envmon
```

The following example shows how to enable the router to send all traps to the host example.com using the community string public:

```
snmp-server enable traps
snmp-server host example.com public
```

The following example shows a configuration in which no traps are sent to a host. The BGP traps are enabled for all hosts, but only the ISDN traps are enabled to be sent to a host.

```
snmp-server enable traps bgp
snmp-server host host1 public isdn
```

The following example shows how to enable a router to send all informs to the host example.com using the community string named public:

```
snmp-server enable traps
snmp-server host example.com informs version 2c public
```

In the following example, the SNMP manager is enabled and the session timeout is set to a value greater than the default:

```
snmp-server manager
snmp-server manager session-timeout 1000
```

Example Configuring IfAlias Long Name Support

In the following example, a long description is applied to the Ethernet interface in slot 1, port adapter 0, and port 0:

```
Router# configure terminal
Router(config)# interface Ethernet1/0/0
Router(config-if)# description ethernet1/0/0 this is a test of a description that exceeds
64 characters in length
Router(config-if)# ip address 192.168.134.55 255.255.255.0
Router(config-if)# no ip directed-broadcast
Router(config-if)# no ip route-cache distributed
```

Assuming that ifAlias long name support is not yet enabled (the default), the following example shows the results of a mibwalk operation from an NMS:

```
***** SNMP QUERY STARTED *****
 .
 .
 .
 ifXEntry.18.10 (octets) (zero-length)
 ifXEntry.18.11 (octets) ethernet1/0/0 this is a test of a description that exceeds 64 ch
 ifXEntry.18.12 (octets) (zero-length)
 .
 .
 .
```

The following output shows the description that is displayed in the CLI:

```
Router# show interface Ethernet0/0/0
Ethernet1/0/0 is administratively down, line protocol is down
  Hardware is Lance, address is 0010.7b4d.7046 (bia 0010.7b4d.7046)
  Description: ethernet1/0/0 this is a test of a description that exceeds 64 chh
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 252/255, txload 1/255, rxload 1/255
 .
 .
 .
```

In the following example, ifAlias long name support is enabled and the description is displayed again:

```
Router(config)# snmp ifmib ifalias long
Router(config)# interface Ethernet1/0/0
```

```
Router(config-if)# description ethernet1/0/0 this is a test of a description that exceeds
64 characters in length
Router(config)# end
Router# show interface Ethernet1/0/0
Ethernet1/0/0 is administratively down, line protocol is down
  Hardware is Lance, address is 0010.7b4d.7046 (bia 0010.7b4d.7046)
  Description: ethernet1/0/0 this is a test of a description that exceeds 64 characters
in length
  MTU 1500 bytes, BW 10000 Kbit, DLY 1000 usec,
     reliability 252/255, txload 1/255, rxload 1/255
 .
 .
 .
***** SNMP QUERY STARTED *****
 .
 .
 .
 ifXEntry.18.10 (octets) (zero-length)
 ifXEntry.18.11 (octets) ethernet1/0/0 this is a test of a description that exceeds 64
characters in length
 ifXEntry.18.12 (octets) (zero-length)
 .
 .
 .
```

Example Configuring IfIndex Persistence

The following example shows how to enable IfIndex persistence globally:

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

The following example shows how to enable IfIndex persistence on the Ethernet interface:

```
Router# configure terminal
Router(config)# interface ethernet 0/1
Router(config)# snmp-server ifindex persist
```

Example Configuring SNMP Support for VPNs

In the following example, all SNMP notifications are sent to example.com over the VRF named trap-vrf:

```
Router(config)# snmp-server host example.com vrf trap-vrf
```

In the following example, the VRF named "traps-vrf" is configured for the remote server 172.16.20.3:

```
Router(config)# snmp-server engineID remote 172.16.20.3 vrf traps-vrf
80000009030000B064EFE100
```

Example Enabling Event MIB Persistence

The following example shows how to enable Event MIB Persistence using the **snmp mib persist event** command in global configuration mode:

```
Router(config)# snmp mib persist event
Router(config)# exit
Router# write mib-data
```

Example Enabling Expression MIB Persistence

The following example shows how to enable Expression MIB Persistence using the **snmp mib persist expression** command in global configuration mode:

```
Router(config)# snmp mib persist expression
Router(config)# exit
Router# write mib-data
```

Example Configuring Event MIB

The following example shows how to configure scalar variables for an event:

```
Router# configure terminal
Router(config)# snmp mib event sample minimum 10
Router(config)# snmp mib event sample instance maximum 50
Router(config)# exit
```

The following example shows how to configure the object list for an event:

```
Router# configure terminal
Router(config)# snmp mib event object list owner owner1 name objectA number 1
Router(config-event-objlist)# object id ifInOctets
Router(config-event-objlist)# wildcard
Router(config-event-objlist)# exit
```

The following example shows how to configure an event:

```
Router# configure terminal
Router(config)# snmp mib event owner owner1 event EventA
Router(config-event)# description "eventA is an RMON event."
Router(config-event)# enable
Router(config-event)# exit
```

The following example shows how to set the notification action for an event:

```
Router(config-event)# action notification
Router(config-event-action-notification)# object id ifInOctets
Router(config-event-action-notification)# exit
```

The following example shows how to set actions for an event:

```
Router(config-event)# action set
Router(config-event-action-set)# object id ifInOctets
Router(config-event-action-set)# value 10
Router(config-event-action-set)# exit
```

The following example shows how to configure the trigger for an event:

```
Router# configure terminal
Router(config)# snmp mib event trigger owner owner1 name EventTriggerA
Router(config-event-trigger)# description EventTriggerA is an RMON alarm.
Router(config-event-trigger)# frequency 120
Router(config-event-trigger)# object list owner owner1 name ObjectListA
Router(config-event-trigger)# object id ifInOctets
Router(config-event-trigger)# enable
Router(config-event-trigger)# exit
```

The following example shows how to configure the existence trigger test:

```
Router(config-event-trigger)# test existence
Router(config-event-trigger-existence)# event owner owner1 name EventA
Router(config-event-trigger-existence)# object list owner owner1 name ObjectListA
Router(config-event-trigger-existence)# type present
Router(config-event-trigger-existence)# startup present
Router(config-event-trigger-existence)# exit
```

The following example shows how to configure the Boolean trigger test:

```
Router(config-event-trigger)# test boolean
Router(config-event-trigger-boolean)# comparison unequal
Router(config-event-trigger-boolean)# value 10
Router(config-event-trigger-boolean)# object list owner owner1 name ObjectListA
Router(config-event-trigger-boolean)# event owner owner1 name EventA
Router(config-event-trigger-boolean)# startup
Router(config-event-trigger-boolean)# exit
```

The following example shows how to configure the threshold trigger test:

```
Router(config-event-trigger)# test threshold
Router(config-event-trigger-threshold)# object list owner owner1 name ObjectListA
Router(config-event-trigger-threshold)# rising 100
Router(config-event-trigger-threshold)# rising event owner owner1 name EventA
Router(config-event-trigger-threshold)# falling 50
```

```
Router(config-event-trigger-threshold)# falling event owner owner1 name EventA
Router(config-event-trigger-threshold)# delta rising 30
Router(config-event-trigger-threshold)# delta rising event owner owner1 name EventA
Router(config-event-trigger-threshold)# delta falling 10
Router(config-event-trigger-threshold)# delta falling event owner owner1 name EventA
Router(config-event-trigger-threshold)# startup rising
Router(config-event-trigger-threshold)# exit
```

Example Configuring Expression MIB

The following example shows how to configure the Expression MIB by using the **snmp mib expression** command in global configuration mode:

```
Router(config)# snmp mib expression owner pcn name exp6
Router(config-expression)# description this expression is created for the sysLocation MIB
object
Router(config-expression)# expression ($1+$2)*800/$3
Router(config-expression)# delta interval 120
Router(config-expression)# value type counter32
Router(config-expression)# enable
Router(config-expression)# object 2
Router(config-expression-object)# id ifInOctets
Router(config-expression-object)# wilcard
Router(config-expression-object)# discontinuity object sysUpTime
Router(config-expression-object)# conditional object
mib-2.90.1.3.1.1.2.3.112.99.110.4.101.120.112.53 wildcard
Router(config-expression-object)# sample delta
Router(config-expression-object)# exit
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	*Cisco IOS Master Commands List, All Releases*
SNMP commands: complete command syntax, command mode, command history, defaults, usage guidelines, and examples	*Cisco IOS SNMP Command Reference*
Cisco IOS implementation of RFC 1724, RIP Version 2 MIB Extensions	*RIPv2 Monitoring with SNMP Using the RFC 1724 MIB Extensions* feature module
DSP Operational State Notifications for notifications to be generated when a digital signaling processor (DSP) is used	*DSP Operational State Notifications* feature module

Standards and RFCs

Standard/RFC	Title
CBC-DES (DES-56) standard	*Symmetric Encryption Protocol*
STD: 58	*Structure of Management Information Version 2 (SMIv2)*

Standard/RFC	Title
RFC 1067	*A Simple Network Management Protocol*
RFC 1091	*Telnet terminal-type option*
RFC 1098	*Simple Network Management Protocol (SNMP)*
RFC 1157	*Simple Network Management Protocol (SNMP)*
RFC 1213	*Management Information Base for Network Management of TCP/IP-based internets:MIB-II*
RFC 1215	*Convention for defining traps for use with the SNMP*
RFC 1901	*Introduction to Community-based SNMPv2*
RFC 1905	*Common Management Information Services and Protocol over TCP/IP (CMOT)*
RFC 1906	*Telnet X Display Location Option*
RFC 1908	*Simple Network Management Protocol (SNMP)*
RFC 2104	*HMAC: Keyed-Hashing for Message Authentication*
RFC 2206	*RSVP Management Information Base using SMIv2*
RFC 2213	*Integrated Services Management Information Base using SMIv2*
RFC 2214	*Integrated Services Management Information Base Guaranteed Service Extensions using SMIv2*
RFC 2271	*An Architecture for Describing SNMP Management Frameworks*
RFC 2570	*Introduction to Version 3 of the Internet-standard Network Management Framework*
RFC 2578	*Structure of Management Information Version 2 (SMIv2)*
RFC 2579	*Textual Conventions for SMIv2*
RFC 2580	*Conformance Statements for SMIv2*
RFC 2981	*Event MIB*
RFC 2982	*Distributed Management Expression MIB*
RFC 3413	*SNMPv3 Applications*

Standard/RFC	Title
RFC 3415	*View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)*

MIBs

MIB	MIBs Link
• Circuit Interface Identification MIB • Cisco SNMPv2 • Ethernet-like Interfaces MIB • Event MIB • Expression MIB Support for Delta, Wildcarding, and Aggregation • Interfaces Group MIB (IF-MIB) • Interfaces Group MIB Enhancements • MIB Enhancements for Universal Gateways and Access Servers • MSDP MIB • NTP MIB • Response Time Monitor MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for Configuring SNMP Support

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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

Table 4 Feature Information for Configuring SNMP Support

Feature Name	Releases	Feature Information
Circuit Interface Identification Persistence for SNMP	12.1(3)T 15.0(1)S	The Circuit Interface Identification Persistence for SNMP feature can be used to identify individual circuit-based interfaces for SNMP monitoring.
Entity MIB, Phase I	11.3(1) 12.0(1) 12.2(2)T 15.0(1)S	The Entity MIB feature implements support for the Entity MIB module, defined in RFC 2037, and provides a mechanism by which a managed device can advertise its logical components, physical components, and logical to physical mappings.
Event MIB	12.0(12)S 12.1(3)T 15.0(1)S	The Event MIB feature provides the ability to monitor Management Information Base (MIB) objects on a local or remote system using SNMP and initiate simple actions whenever a trigger condition is met. By allowing notifications based on events, the Network Management Server (NMS) does not need to constantly poll managed devices to find out if something has changed.

Feature Name	Releases	Feature Information
Event MIB and Expression MIB CLIs	12.2(33)SRE 12.2(50)SY 12.4(20)T 15.0(1)S	The Event MIB and Expression MIB feature introduces CLIs to configure the Event MIB and Expression MIB. The following commands were introduced or modified by this feature: **action (event)**, **comparison**, **conditional object**, **delta (test threshold)**, **delta interval**, **description (event)**, **description (expression)**, **description (trigger)**, **discontinuity object (expression)**, **enable (event)**, **enable (expression)**, **event owner**, **expression**, **falling (test threshold)**, **frequency (event trigger)**, **object (expression)**, **object id**, **object list**, **rising (test threshold)**, **sample (expression)**, **snmp mib event object list**, **snmp mib event owner**, **snmp mib event trigger owner**, **snmp mib expression delta**, **snmp mib expression owner**, **startup (test boolean)**, **startup (test existence)**, **startup (test threshold)**, **test (event trigger)**, **type (test existence)**, **value (test boolean)**, **value type**, **wildcard (expression)**.
Expression MIB Support of Delta, Wildcarding and Aggregation	12.1(3)T 15.0(1)S	The Expression MIB Support of Delta, Wildcarding and Aggregation feature adds support of Delta, Wildcarding, and Aggregation to the Expression MIB implementation.

Feature Name	Releases	Feature Information
Interface Index Display for SNMP	12.2(2)T	The Interface Index Display for SNMP feature introduces new commands and command modifications that allow advanced users of SNMP to view information about interface registrations directly on the managed agent. You can display MIB information from the agent without using an external NMS. This feature addresses three objects in the Interfaces MIB: *ifIndex*, *ifAlias*, and *ifName*. For complete definitions of these objects, see the IF-MIB.my file available at the Cisco SNMPv2 MIB website: ftp://ftp.cisco.com/pub/mibs/v2/.
Interface Index Persistence	12.2(15)T 15.0(1)S	The Interface Index Persistence feature allows interfaces to be identified with unique values, which will remain constant even when a device is rebooted. These interface identification values are used for network monitoring and management using SNMP.
Interfaces MIB: SNMP context based access	12.2(33)SRB 12.2(33)SB 12.2(44)SG 15.0(1)S	The Interfaces MIB: SNMP context based access feature provides the ability to query Interfaces MIB objects. The information returned will be restricted to the VRF to which the SNMP context is mapped. Notification hosts may also be configured with contexts to restrict notifications that need to be sent to the particular host.

Feature Name	Releases	Feature Information
MIB Persistence	12.0(5)T 12.0(12)S 12.1(3)T 12.2(4)T 12.2(4)T3	The MIB Persistence feature allows the SNMP data of a MIB to be persistent across reloads; this means MIB information retains the same set object values each time a networking device reboots. MIB Persistence is enabled by using the **snmp mib persist** command, and the MIB data of all MIBs that have had persistence enabled using this command is then written to NVRAM storage by using the **write mib-data** command. Any modified MIB data must be written to the NVRAM memory using the **write mib-data** command.
SNMP (Simple Network Management Protocol)	11.2(1) 15.0(1)S	The Simple Network Management Protocol (SNMP) feature provides an application-layer protocol that facilitates the exchange of management information between network devices. It is part of the TCP/IP protocol suite. SNMP enables network administrators to manage network performance, find and solve network problems, and plan for network growth.
SNMP Diagnostics	12.2(33)SRE 12.2(50)SY 12.4(20)T 15.0(1)S	The SNMP Diagnostics feature adds Cisco IOS CLI commands to display object identifiers recently requested by the network management system, and to display the SNMP debug messages. The following commands were introduced or modified: **debug snmp detail**, **show snmp stats oid**.

Feature Name	Releases	Feature Information
SNMP Inform Request	11.3(1)T 12.0(1)T 12.1(3)T 12.1(14) 12.2(8)T 15.0(1)S	The SNMP Inform Request feature supports sending inform requests. SNMP asynchronous notifications are usually sent as SNMP traps. Traps are less reliable than informs because an acknowledgment is not sent from the receiving end when a trap is received; however, an SNMP manager that receives an inform acknowledges the message with an SNMP response PDU. If the sender does not receive a response for an inform, the inform can be sent again.
SNMP Manager	11.3(1) 11.3(1)T 12.0(1) 15.0(1)S	The SNMP Manager feature adds a system that controls and monitors the activities of network hosts using SNMP. The most common managing system is an NMS.
SNMP Notification Logging	12.0(22)S 12.2(13)T	The SNMP Notification Logging feature adds Cisco IOS CLI commands to change the size of the notification log, set the global ageout value for the log, and display logging summaries at the command line.
SNMP Support for VPNs	12.0(23)S 12.2(2)T 12.2(33)SB 12.2(33)SXH 15.0(1)S Cisco IOS XE Release 3.1.0SG	The SNMP Support for VPNs feature allows SNMP traps and informs to be sent and received using VRF tables. In particular, this feature adds support to the Cisco IOS software for sending and receiving SNMP traps and informs specific to individual VPNs.
SNMP Trap Simulations	12.2(33)SRE 12.2(33)SXI 15.0(1)S	The SNMP Trap Simulations feature introduces the **test snmp trap** commands to verify the reception of SNMP, syslog, and config-copy notifications by the SNMP manager in a simulated scenario.

Feature Name	Releases	Feature Information
SNMP Version 2 (SNMPv2)	11.3(1) 12.0(1) 15.0(1)S	The SNMP Version 2 (SNMPv2) feature represents the community string-based Administrative Framework for SNMPv2. SNMPv2c support includes a bulk retrieval mechanism and detailed error message reporting to management stations.
SNMP Version 3 (SNMPv3)	12.0(3)T 12.0(6)S 12.1(3)T 12.1(14) 12.2(13)T 15.0(1)S	The SNMP Version 3 (SNMPv3) feature provides support for USM in which an authentication strategy is set up for a user and the group in which the user resides.
SNMPv3 Community MIB Support	12.0(22)S 12.2(4)T 12.2(11)T 12.2(18)S 15.0(1)S Cisco IOS XE Release 3.1.0SG	The SNMPv3 Community MIB Support feature implements support for the SNMP Community MIB (SNMP-COMMUNITY-MIB) module, defined in RFC 2576, in the Cisco IOS software.

Glossary

ifAlias—SNMP Interface Alias. The ifAlias is an object in the IF-MIB. The ifAlias is an alias name for the interface as specified by the network manager that provides a nonvolatile description for the interface. For a complete definition, see the IF-MIB.my file.

ifIndex—SNMP Interface Index. The ifIndex is an object in the IF-MIB. The ifIndex is a unique integer assigned to every interface (including subinterfaces) on the managed system when the interface registers with the IF-MIB. For a complete definition, see the IF-MIB.my file.

OID—MIB object identifier. An object identifier is expressed as a series of integers or text strings. Technically, the numeric form is the *object name* and the text form is the *object descriptor*. In practice, both are called object identifiers or OIDs. For example, the object name for the interfaces MIB is 1.3.6.1.2.1.2, and the object descriptor is ‘iso.internet.mgmt.mib-2.interfaces’, but either can be referred to as the OID. An OID can also be expressed as a combination of the two, such as iso.internet.2.1.2.

RIPv2 Monitoring with SNMP Using the RFC 1724 MIB Extensions

This document describes the Cisco IOS implementation of RFC 1724, *RIP Version 2 MIB Extensions* . RFC 1724 defines Management Information Base (MIB) objects that allow you to monitor RIPv2 using Simple Network Management Protocol (SNMP).

Finding Feature Information in This Module

Your Cisco IOS software release may not support all of the features documented in this module. To reach links to specific feature documentation in this module and to see a list of the releases in which each feature is supported, use the Feature Information for RIPv2 RFC 1724 MIB Extensions, page 119.

Finding Support Information for Platforms and Cisco Software Images

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.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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

Prerequisites for RIPv2 Monitoring with SNMP Using the RFC 1724 MIB Extensions

- RIPv2 must be configured on the router.
- Your SNMP Network Management Station (NMS) must have the RFC 1724 RIPv2 MIB installed.
- Your SNMP NMS must have the following MIBs installed because RFC 1724 imports data types and object Identifiers (OIDs) from them:
  - SNMPv2-SMI
  - SNMPv2-TC
  - SNMPv2-CONF
  - RFC1213-MIB

Restrictions for RIPv2 Monitoring with SNMP Using the RFC 1724 MIB Extensions

This implementation of the RIPv2 MIB does not track any data associated with a RIP Virtual Routing and Forwarding (VRF) instance. Only interfaces that are assigned IP addresses in the IP address space configured by the **network** *network-address* command in RIP router configuration mode are tracked. Global data is tracked only for changes to the main routing table.

Information About RIPv2 Monitoring with SNMP Using the RFC 1724 MIB Extensions

The following sections contain information about MIB objects standardized as part of RFC 1724, and benefits of the RFC 1724 MIB.

- RIPv2 MIB, page 110
- Benefits of the RIPv2 MIB, page 113

RIPv2 MIB

This section describes the MIB objects that are provided by RFC 1724 definitions. The RIPv2 MIB consists of the following managed objects:

- Global counters--Used to keep track of changing routes or neighbor changes.
- Interface status table--Defines objects that are used to keep track of statistics specific to interfaces.
- Interface configuration table--Defines objects that are used to keep track of interface configuration statistics.
- Peer table--Defined to monitor neighbor relationships. This object is not implemented in Cisco IOS Software.

The tables below show the objects that are provided by RFC 1724 RIPv2 MIB definitions. The objects are listed in the order in which they appear within the RFC 1724 RIPv2 MIB, per the tables that describe them.

The statistics for all of the objects in the global counters can be obtained by querying the rip2Globals object identifier (OID) using **snmpwalk**, or a similar SNMP toolset command on your NMS.

The table below shows the RFC 1724 RIPv2 MIB global counter objects.

Table 5 RFC 1724 RIPv2 MIB Global Counters Objects

Global Counter	Object	Description
rip2Globals	rip2GlobalRouteChanges	Number of route changes made to the IP route database by RIP. Number is incremented when a route is modified.
	rip2GlobalQueries	Number of responses sent to RIP queries from other systems. Number is incremented when RIP responds to a query from another system.

The objects in the RFC 1724 RIPv2 MIB interface table track information on a per interface basis. All object in the RFC 1724 RIPv2 MIB interface table, except for the rip2IfStatAddress object, represent newly tracked data within RIP. There are no equivalent **show** commands for these objects. All objects in the RIPv2 MIB interface table are implemented read-only.

The table below shows the RFC 1724 RIPv2 MIB interface table objects. The statistics for all objects in the interface table can be obtained by querying the sequence name Rip2IfStatEntry using **snmpwalk** or a similar SNMP toolset command on your NMS.

Table 6 RFC 1724 RIPv2 MIB Interface Table Objects

Sequence Name	Object	Description
Rip2IfStatEntry	rip2IfStatAddress	The IP address of this system on the indicated subnet. For unnumbered interfaces, the value of 0.0.0.N, where the least significant 24 bits (N) are the ifIndex for the IP interface in network byte order.
	rip2IfStatRcvBadPackets	The number of RIP response packets received by the RIP process that were subsequently discarded for any reason. For example, a version 0 packet or an unknown command type.

Sequence Name	Object	Description
	rip2IfStatRcvBadRoutes	The number of routes, in valid RIP packets, that were ignored for any reason. This is incremented when: • The address family identifier does not equal AF_INET. • If a RIP v2 update is received and the class D and greater. • If a RIP v2 update is received and the address is a martian address.
	rip2IfStatSentUpdates	The number of triggered RIP updates actually sent on this interface. This explicitly does *not* include full updates sent containing new information.
	rip2IfStatStatus	This value is always set to 1.

The objects in the RFC 1724 RIPv2 MIB interface configuration table track information on a per interface basis. Except for the Rip2IfConfAuthType object, the data for the objects in the RFC 1724 RIPv2 MIB interface configuration table can also be gathered with the **show ip protocol** commands. All objects in the RIPv2 MIB interface table are implemented read-only.

The table below shows the RIPv2 MIB interface configuration table objects. The statistics for all objects in the configuration table can be obtained by querying the sequence name rip2IfConfEntry using **snmpwalk** or a similar SNMP toolset command on your NMS.

Table 7 RFC 1724 RIPv2 MIB Interface Configuration Table Object Types

Sequence Name	Object Type	Description
rip2IfConfEntry	rip2IfConfAddress	The IP address of this system on the indicated subnet. For unnumbered interfaces, the value 0.0.0.N, where the least significant 24 bits (N) are the ifIndex for the IP interface in network byte order.
	rip2IfConfDomain	This value is always equal to "".
	rip2IfConfAuthType	The type of authentication used on this interface.

Sequence Name	Object Type	Description
	rip2IfConfAuthKey	The value to be used as the authentication key whenever the corresponding instance of rip2IfConfAuthType has a value other than no authentication.
	rip2IfConfSend	The version of RIP updates that are sent on this interface.
	rip2IfConfReceive	The version of RIP updates that are accepted on this interface.
	rip2IfConfDefaultMetric	This variable indicates the metric that is used for the default route entry in RIP updates originated on this interface.
	rip2IfConfStatus	This value is always set to 1.
	rip2IfConfSrcAddress	The IP address that this system will use as a source address on this interface. If it is a numbered interface, this *must* be the same value as rip2IfConfAddress. On unnumbered interfaces, it must be the value of rip2IfConfAddress for some interface on the system.

Benefits of the RIPv2 MIB

The RFC 1724 RIPv2 MIB extensions allow network managers to monitor the RIPv2 routing protocol using SNMP through the addition of new global counters and table objects that previously were not supported by the RFC 1389 RIPv2 MIB. The new global counters and table objects are intended to facilitate quickly changing routes or failing neighbors.

How to Enable RIPv2 Monitoring with SNMP Using the RIPv2 RFC124 MIB Extensions

Enabling SNMP Read-Only Access on the Router

There are no router configuration tasks required for the RIPv2: RFC124 MIB Extensions feature itself. SNMP read-only access to the objects in the RFC 1724 RIPv2 MIB is enabled when you configure the SNMP server read-only community string on the router.

Note When you configure an SNMP server read-only community string on the router, you are granting SNMP read-only access to the objects that support read-only access in all MIBs that are available in the version of Cisco IOS that is running on the router.

Perform this task to configure the SNMP server read-only community string on the router to enable SNMP read-only access to MIB objects (including the RFC 1724 RIPv2 MIB extensions) on the router.

Routers can have multiple read-only SNMP community strings. When you configure an SNMP read-only community string for the **snmp-server** command on the router, an existing SNMP **snmp-server** read-only community string is not overwritten. For example, if you enter the **snmp-server community** *string1* **ro** and **snmp-server community** *string2* **ro commands** on the router,the router will have two valid read-only community strings--*string1* and *string2*. If this is not the behavior that you desire, use the **no snmp-server community** *string* **ro** command to remove an existing SNMP read-only community string.

Timesaver If you already have an SNMP read-only community string configured on your router you do not need to perform this task. After you load Cisco IOS Release 12.4(6)T or a later release on your router, you can use SNMP commands on your NMS to query the RFC 1724 RIPv2 MIB on your router.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server community** *string1* **ro**
4. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server community** *string1* **ro** **Example:** `Router(config)# snmp-server community T8vCx3 ro`	Enables SNMP read-only access to the objects in the MIBs that are included in the version of Cisco IOS software that is running on the router. **Note** For security purposes, do not use the standard default value of *public* for your read-only community string. Use a combination of uppercase and lowercase letters and numbers for the password.
Step 4	**end** **Example:** `Router(config)# end`	Ends your configuration session and returns to privileged EXEC mode.

Verifying the Status of the RIPv2 RFC124 MIB Extensions on the Router and Your Network Management Station

Perform this optional task on your NMS to verify the status of the RFC 1724 RIPv2 MIB extensions on the router and on your NMS.

Note This task uses the NET-SNMP toolset that is available in the public domain. The step that is documented uses a terminal session on an NMS that is running Linux. Substitute the SNMP command from the SNMP toolset on your NMS as appropriate when you perform this task.

- Prerequisites, page 115

Prerequisites

Your NMS must have the RFC 1724 MIB installed.

SUMMARY STEPS

1. **snmpwalk -m all -v2c** *ip-address* **-c** *read-only-community-string* **rip2Globals**

DETAILED STEPS

snmpwalk -m all -v2c *ip-address* **-c** *read-only-community-string* **rip2Globals**
Use the **snmpwalk** command for the **rip2Globals** object in the RFC 1724 RIPv2 MIB to display the data for the objects associated with this object. This step verifies that the NMS is configured to send queries for objects in the RFC 1724 RIPv2 MIB and that the router is configured to respond to the queries.

Example:

```
$ snmpwalk -m all -v2c 10.0.0.253 -c T8vCx3 rip2Globals
```

```
RIPv2-MIB::rip2GlobalRouteChanges.0 = Counter32: 5
RIPv2-MIB::rip2GlobalQueries.0 = Counter32: 1
$
```

Configuration Examples for RIPv2 Monitoring with SNMP Using the RIPv2 RFC124 MIB Extensions

This section contains the following examples:

Querying the RIP Interface Status Table Objects Example

The following example shows how to send an SNMP query to obtain data for all objects in the RIP interface status table using the **snmpwalk** command.

```
$ snmpwalk -m all -v2c 10.0.0.253 -c T8vCx3 Rip2IfStatEntry
RIPv2-MIB::rip2IfStatAddress.10.0.0.253 = IpAddress: 10.0.0.253
RIPv2-MIB::rip2IfStatAddress.172.16.1.1 = IpAddress: 172.16.1.1
RIPv2-MIB::rip2IfStatAddress.172.16.2.1 = IpAddress: 172.16.2.1
RIPv2-MIB::rip2IfStatAddress.172.17.1.1 = IpAddress: 172.17.1.1
RIPv2-MIB::rip2IfStatAddress.172.17.2.1 = IpAddress: 172.17.2.1
RIPv2-MIB::rip2IfStatRcvBadPackets.10.0.0.253 = Counter32: 0
RIPv2-MIB::rip2IfStatRcvBadPackets.172.16.1.1 = Counter32: 1654
RIPv2-MIB::rip2IfStatRcvBadPackets.172.16.2.1 = Counter32: 1652
RIPv2-MIB::rip2IfStatRcvBadPackets.172.17.1.1 = Counter32: 1648
RIPv2-MIB::rip2IfStatRcvBadPackets.172.17.2.1 = Counter32: 1649
RIPv2-MIB::rip2IfStatRcvBadRoutes.10.0.0.253 = Counter32: 0
RIPv2-MIB::rip2IfStatRcvBadRoutes.172.16.1.1 = Counter32: 0
RIPv2-MIB::rip2IfStatRcvBadRoutes.172.16.2.1 = Counter32: 0
RIPv2-MIB::rip2IfStatRcvBadRoutes.172.17.1.1 = Counter32: 0
RIPv2-MIB::rip2IfStatRcvBadRoutes.172.17.2.1 = Counter32: 0
RIPv2-MIB::rip2IfStatSentUpdates.10.0.0.253 = Counter32: 0
RIPv2-MIB::rip2IfStatSentUpdates.172.16.1.1 = Counter32: 0
RIPv2-MIB::rip2IfStatSentUpdates.172.16.2.1 = Counter32: 0
RIPv2-MIB::rip2IfStatSentUpdates.172.17.1.1 = Counter32: 0
RIPv2-MIB::rip2IfStatSentUpdates.172.17.2.1 = Counter32: 0
RIPv2-MIB::rip2IfStatStatus.10.0.0.253 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.16.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.16.2.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.17.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.17.2.1 = INTEGER: active(1)
```

The following example shows how to send an SNMP query to obtain data for the rip2IfStatStatus object for all of the interfaces in the RIP interface status table using the **snmpwalk** command.

```
$ snmpwalk -m all -v2c 10.0.0.253 -c T8vCx3 rip2IfStatStatus
RIPv2-MIB::rip2IfStatStatus.10.0.0.253 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.16.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.16.2.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.17.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfStatStatus.172.17.2.1 = INTEGER: active(1)
$
```

The following example shows how to send an SNMP query to obtain data for the rip2IfStatStatus object for a specific interface IP address in the RIP interface status table using the **snmpget** command.

```
$ snmpget -m all -v2c 10.0.0.253 -c T8vCx3 rip2IfStatStatus.10.0.0.253
RIPv2-MIB::rip2IfStatStatus.10.0.0.253 = INTEGER: active(1)
$
```

Querying the RIP Interface Configuration Table Objects Example

The following example shows how to send an SNMP query to obtain data for all objects in the RIP interface configuration table using the **snmpwalk** command.

```
$ snmpwalk -m all -v2c 10.0.0.253 -c T8vCx3 rip2IfConfEntry
RIPv2-MIB::rip2IfConfAddress.10.0.0.253 = IpAddress: 10.0.0.253
RIPv2-MIB::rip2IfConfAddress.172.16.1.1 = IpAddress: 172.16.1.1
RIPv2-MIB::rip2IfConfAddress.172.16.2.1 = IpAddress: 172.16.2.1
RIPv2-MIB::rip2IfConfAddress.172.17.1.1 = IpAddress: 172.17.1.1
RIPv2-MIB::rip2IfConfAddress.172.17.2.1 = IpAddress: 172.17.2.1
RIPv2-MIB::rip2IfConfDomain.10.0.0.253 = ""
RIPv2-MIB::rip2IfConfDomain.172.16.1.1 = ""
RIPv2-MIB::rip2IfConfDomain.172.16.2.1 = ""
RIPv2-MIB::rip2IfConfDomain.172.17.1.1 = ""
RIPv2-MIB::rip2IfConfDomain.172.17.2.1 = ""
RIPv2-MIB::rip2IfConfAuthType.10.0.0.253 = INTEGER: noAuthentication(1)
RIPv2-MIB::rip2IfConfAuthType.172.16.1.1 = INTEGER: noAuthentication(1)
RIPv2-MIB::rip2IfConfAuthType.172.16.2.1 = INTEGER: noAuthentication(1)
RIPv2-MIB::rip2IfConfAuthType.172.17.1.1 = INTEGER: noAuthentication(1)
RIPv2-MIB::rip2IfConfAuthType.172.17.2.1 = INTEGER: noAuthentication(1)
RIPv2-MIB::rip2IfConfAuthKey.10.0.0.253 = ""
RIPv2-MIB::rip2IfConfAuthKey.172.16.1.1 = ""
RIPv2-MIB::rip2IfConfAuthKey.172.16.2.1 = ""
RIPv2-MIB::rip2IfConfAuthKey.172.17.1.1 = ""
RIPv2-MIB::rip2IfConfAuthKey.172.17.2.1 = ""
RIPv2-MIB::rip2IfConfSend.10.0.0.253 = INTEGER: ripVersion2(4)
RIPv2-MIB::rip2IfConfSend.172.16.1.1 = INTEGER: ripVersion2(4)
RIPv2-MIB::rip2IfConfSend.172.16.2.1 = INTEGER: ripVersion2(4)
RIPv2-MIB::rip2IfConfSend.172.17.1.1 = INTEGER: ripVersion2(4)
RIPv2-MIB::rip2IfConfSend.172.17.2.1 = INTEGER: ripVersion2(4)
RIPv2-MIB::rip2IfConfReceive.10.0.0.253 = INTEGER: rip2(2)
RIPv2-MIB::rip2IfConfReceive.172.16.1.1 = INTEGER: rip2(2)
RIPv2-MIB::rip2IfConfReceive.172.16.2.1 = INTEGER: rip2(2)
RIPv2-MIB::rip2IfConfReceive.172.17.1.1 = INTEGER: rip2(2)
RIPv2-MIB::rip2IfConfReceive.172.17.2.1 = INTEGER: rip2(2)
RIPv2-MIB::rip2IfConfDefaultMetric.10.0.0.253 = INTEGER: 1
RIPv2-MIB::rip2IfConfDefaultMetric.172.16.1.1 = INTEGER: 1
RIPv2-MIB::rip2IfConfDefaultMetric.172.16.2.1 = INTEGER: 1
RIPv2-MIB::rip2IfConfDefaultMetric.172.17.1.1 = INTEGER: 1
RIPv2-MIB::rip2IfConfDefaultMetric.172.17.2.1 = INTEGER: 1
RIPv2-MIB::rip2IfConfStatus.10.0.0.253 = INTEGER: active(1)
RIPv2-MIB::rip2IfConfStatus.172.16.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfConfStatus.172.16.2.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfConfStatus.172.17.1.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfConfStatus.172.17.2.1 = INTEGER: active(1)
RIPv2-MIB::rip2IfConfSrcAddress.10.0.0.253 = IpAddress: 10.0.0.253
RIPv2-MIB::rip2IfConfSrcAddress.172.16.1.1 = IpAddress: 172.16.1.1
RIPv2-MIB::rip2IfConfSrcAddress.172.16.2.1 = IpAddress: 172.16.2.1
RIPv2-MIB::rip2IfConfSrcAddress.172.17.1.1 = IpAddress: 172.17.1.1
RIPv2-MIB::rip2IfConfSrcAddress.172.17.2.1 = IpAddress: 172.17.2.1
$
```

The following example shows how to send an SNMP query to obtain data for the rip2IfConfAddress object for all interfaces in the RIP interface configuration table using the **snmpwalk** command.

```
$ snmpwalk -m all -v2c 10.0.0.253 -c T8vCx3 rip2IfConfAddress
RIPv2-MIB::rip2IfConfAddress.10.0.0.253 = IpAddress: 10.0.0.253
RIPv2-MIB::rip2IfConfAddress.172.16.1.1 = IpAddress: 172.16.1.1
RIPv2-MIB::rip2IfConfAddress.172.16.2.1 = IpAddress: 172.16.2.1
RIPv2-MIB::rip2IfConfAddress.172.17.1.1 = IpAddress: 172.17.1.1
```

```
RIPv2-MIB::rip2IfConfAddress.172.17.2.1 = IpAddress: 172.17.2.1
$
```

Where to Go Next

For more information about SNMP and SNMP operations, see the “Configuring SNMP Support” chapter of the Cisco IOS Network Management Configuration Guide, Release 12.4.

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
RIP configuration	*Cisco IOS IP Routing Protocols Configuration Guide , Release* 12.4
RIP commands	*Cisco IOS IP Routing Protocols Configuration Guide , Release* 12.4T
SNMP configuration	*Cisco IOS Network Management Configuration Guide , Release* 12.4
SNMP commands	*Cisco IOS Network Management Command Reference*, Release 12.4T

Standards

Standard	Title
No new or modified standards are supported by this feature, and support for existing standards has not been modified by this feature.	--

MIBs

MIB	MIBs Link
RIPv2 MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1724	*RIP Version 2 MIB Extensions*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for RIPv2 RFC 1724 MIB Extensions

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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

Table 8 Feature Information for RIPv2: RFC 1724 MIB Extensions

Feature Name	Releases	Feature Information
RIPv2: RFC 1724 MIB Extension	12.4(6)T	This feature introduces the Cisco IOS implementation of RFC 1724, *RIP Version 2 MIB Extensions* . RFC 1724 defines MIB objects that allow the management and limited control of RIPv2 using SNMP.

Glossary

OID --object identifier, A managed object within the object tree.

SNMP --Simple Network Management Protocol, a protocol used to monitor and manage networking devices.

snmpwalk --An SNMP command to query statistics from a branch in the MIB.

snmpget --An SNMP command to query statistics from a specific OID in the MIB.

Note See *Internetworking Terms and Acronyms* for terms not included in this glossary.

SNMP Support over VPNs--Context-Based Access Control

The SNMP Support over VPNs--Context-Based Access Control feature provides the infrastructure for multiple Simple Network Management Protocol (SNMP) context support in Cisco IOS software and VPN-aware MIB infrastructure using the multiple SNMP context support infrastructure.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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

Restrictions for SNMP Support over VPNs--Context-Based Access Control

- If you delete an SNMP context using the **no snmp-server context** command, all SNMP instances in that context are deleted.
- Not all MIBs are VPN-aware.

Information About SNMP Support over VPNs--Context-Based Access Control

SNMP Versions and Security

Cisco IOS software supports the following versions of SNMP:

- SNMPv1--Simple Network Management Protocol: a full Internet standard, defined in RFC 1157. (RFC 1157 replaces the earlier versions that were published as RFC 1067 and RFC 1098.) Security is based on community strings.
- SNMPv2c--The community string-based Administrative Framework for SNMPv2. SNMPv2c (the "c" is for "community") is an experimental Internet protocol defined in RFC 1901, RFC 1905, and RFC 1906. SNMPv2c is an update of the protocol operations and data types of SNMPv2p (SNMPv2 Classic) and uses the community-based security model of SNMPv1.
- SNMPv3--Version 3 of SNMP. SNMPv3 is an interoperable standards-based protocol defined in RFCs 3413 to 3415. SNMPv3 provides secure access to devices by authenticating and encrypting packets over the network.

For more information about SNMP Versions, see the “ Configuring SNMP Support ” module in the *Cisco IOS Network Management Configuration Guide* .

SNMPv1 or SNMPv2 Security

SNMPv1 and SNMPv2 are not as secure as SNMPv3. SNMP version 1 and 2 use plain text communities and do not perform the authentication or security checks that SNMP version 3 performs. To configure the SNMP Support over VPNs--Context-Based Access Control feature when using SNMP version 1 or SNMP version 2, you need to associate a community name with a VPN. This association causes SNMP to process requests coming in for a particular community string only if it comes in from the configured VRF. If the community string contained in the incoming packet does not have an associated VRF, it is processed only if it came in through a non-VRF interface. This process prevents users outside the VPN from snooping a clear text community string to query the VPN’s data. These methods of source address validation are not as secure as using SNMPv3.

SNMPv3 Security

If you are using SNMPv3, the security name should always be associated with authentication or privileged passwords. Source address validation is not performed on SNMPv3 users. To ensure that a VPN’s user has access only to context associated to the VPN and cannot see the MIB data of other VPNs, you must configure a minimum security level of AuthNoPriv.

On a provider edge (PE) router, a community can be associated with a VRF to provide source address validation. However, on a customer edge (CE) router, if source address validation is to be provided, you must associate a source address with the community list by using an access control list.

If you are using SNMPv3, the security name or security password of the users of a VPN should be unknown to users of other VPNs. Cisco recommends not to use SNMPv3 nonauthorized users if you need security of management information.

SNMP Notification Support over VPNs

The SNMP Notification Support over VPNs feature allows the sending and receiving of SNMP notifications (traps and informs) using VPN routing and forwarding (VRF) instance tables. In particular, this feature adds support to Cisco IOS software for the sending and receiving of SNMP notifications (traps and informs) specific to individual VPNs.

SNMP is an application-layer protocol that provides a message format for communication between SNMP managers and agents.

A VPN is a network that provides high-connectivity transfers on a shared system with the same usage guidelines as a private network. A VPN can be built on the Internet over IP, Frame Relay, or ATM networks.

A VRF stores per-VPN routing data. It defines the VPN membership of a customer site attached to the network access server (NAS). A VRF consists of an IP routing table, a derived Cisco Express Forwarding (formerly known as CEF) table, and guidelines and routing protocol parameters that control the information that is included in the routing table.

The SNMP Support for VPNs--Context-Based Access Control feature provides configuration commands that allow users to associate SNMP agents and managers with specific VRFs. The associated VRF is used for the sending of SNMP notifications (traps and informs) and responses between agents and managers. If a VRF is not specified, the default routing table for the VPN is used.

VPN-Aware SNMP

The SNMP Support for VPNs--Context-Based Access Control feature extends the capabilities of the SNMP Notification Support for VPNs feature and enables SNMP to differentiate between incoming packets from different VPNs.

When the SNMP Support for VPNs--Context-Based Access Control feature is configured, SNMP accepts requests on any configured VRF and returns responses to the same VRF. A trap host also can be associated with a specific VRF. The configured VRF is then used for sending out traps; otherwise, the default routing table is used. You also can associate a remote user with a specific VRF. You also can configure the VRFs from which SNMP should accept requests. Any requests coming from VRFs that are not specified are dropped.

IP access lists can be configured and associated with SNMP community strings. This feature enables you to configure an association between VRF instances with SNMP community strings. When a VRF instance is associated with an SNMP community string, SNMP processes the requests coming in for a particular community string only if the requests are received from the configured VRF. If the community string contained in the incoming packet does not have a VRF associated with it, the community string will be processed only if it came in through a non-VRF interface.

You also can enable or disable authentication traps for SNMP packets dropped due to VRF mismatches. By default if SNMP authentication traps are enabled, VRF authentication traps are also enabled.

- VPN Route Distinguishers, page 124

VPN Route Distinguishers

A route distinguisher (RD) creates routing and forwarding tables and specifies the default route distinguisher for a VPN. The RD is added to the beginning of the customer's IPv4 prefixes to change them into globally unique VPN-IPv4 prefixes.

The RD is either an autonomous system number (ASN)-relative RD, in which case it comprises an autonomous system number and an arbitrary number, or it is an IP-address-relative RD, in which case it comprises an IP address and an arbitrary number.

You can enter an RD in either of these formats:

- 16-bit ASN: your 16-bit number: For example, 101:3.
- 32-bit IP address: your 32-bit number: For example, 192.168.122.15:1.

SNMP Contexts

SNMP contexts provide VPN users with a secure way of accessing MIB data. When a VPN is associated with a context, that VPN's specific MIB data exists in that context. Associating a VPN with a context enables service providers to manage networks with multiple VPNs. Creating and associating a context with a VPN enables a provider to prevent the users of one VPN from accessing information about other VPN userss on the same networking device.

VPN-aware SNMP requires an agreement between SNMP manager and agent entities operating in a VPN environment on a mapping between the SNMP security name and the VPN ID. This mapping is created by using multiple contexts for the SNMP data of different VPNs through the configuration of the SNMP-VACM-MIB. The SNMP-VACM-MIB is configured with views so that a user on a VPN with a security name is allowed access to the restricted object space associated with a user's access type in the context associated with the user of that VPN.

SNMP request messages undergo three phases of security and access control before a response message is sent back with the object values in the context of a VPN:

- In the first phase, the username is authenticated. This phase ensures that the user is authenticated and authorized for SNMP access.
- In the second phase, the user is authorized for the SNMP access requested to the group objects under consideration of the configured SNMP context. This phase is called the access control phase.
- In the third phase, access is made to a particular instance of a table entry. With this third phase, complete retrieval can be based on the SNMP context name.

How to Configure SNMP Support over VPNs--Context-Based Access Control

- Configuring an SNMP Context and Associating the SNMP Context with a VPN, page 125
- Configuring SNMP Support and Associating an SNMP Context, page 126

Configuring an SNMP Context and Associating the SNMP Context with a VPN

Perform this task to configure an SNMP context and to associate the SNMP context with a VPN.

Note

- Only the following MIBs are context-aware. All the tables in these MIBs can be polled:
  - CISCO-IPSEC-FLOW-MONITOR-MIB (Cisco IOS Release 12.4T and later releases)
  - CISCO-IPSEC-MIB (Cisco IOS Release 12.4T and later releases)
  - CISCO-PING-MIB
  - IP-FORWARD-MIB
  - MPLS-LDP-MIB
- Only two SNMP variables in the IP-FORWARD-MIB can be polled: 1.3.6.1.2.1.4.24.3 (ipCidrRouteNumber - Scalar) and 1.3.6.1.2.1.4.24.4.1 (ipCidrRouteEntry - Table).

>

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server context** *context-name*
4. **ip vrf** *vrf-name*
5. **rd** *route-distinguisher*
6. **context** *context-name*
7. **route-target** {**import** | **export** | **both**} *route-target-ext-community*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

	Command or Action	Purpose
Step 3	**snmp-server context** *context-name* **Example:** `Router(config)# snmp-server context context1`	Creates and names an SNMP context.
Step 4	**ip vrf** *vrf-name* **Example:** `Router(config)# ip vrf vrf1`	Configures a VRF routing table and enters VRF configuration mode.
Step 5	**rd** *route-distinguisher* **Example:** `Router(config-vrf)# rd 100:120`	Creates a VPN route distinguisher.
Step 6	**context** *context-name* **Example:** `Router(config-vrf)# context context1`	Associates an SNMP context with a particular VRF. **Note** In Cisco IOS Release 15.0(1)M and later releases, the **context** command is replaced by the **snmp context** command. See the Network Management Command Reference for more information.
Step 7	**route-target** {**import** | **export** | **both**} *route-target-ext-community* **Example:** `Router(config-vrf)# route-target export 100:1000`	(Optional) Creates a route-target extended community for a VRF.

Configuring SNMP Support and Associating an SNMP Context

Perform this task to configure SNMP support and associate it with an SNMP context.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*] [**vrf** *vrf-name*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes** {**128** | **192** | **256**}} *privpassword*] {*acl-number* | *acl-name*}]
4. **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*] [*acl-number*| *acl-name*]]
5. **snmp-server view** *view-name oid-tree* {**included** | **excluded**}
6. **snmp-server enable traps** [*notification-type*] [**vrrp**]
7. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number* | *extended-access-list-number* | *access-list-name*]
8. **snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*]
9. **snmp mib community-map** *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*][**target-list** *upn-list-name*]
10. **snmp mib target list** *vpn-list-name* {**vrf** *vrf-name* | **host** *ip-address*}
11. **no snmp-server trap authentication vrf**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 3	**snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*] [**vrf** *vrf-name*]] {**v1**	**v2c**	**v3** [**encrypted**] [**auth** {**md5**	**sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des**	**3des**	**aes** {**128**	**192**	**256**}} *privpassword*] {*acl-number*	*acl-name*}] **Example:** `Router(config)# snmp-server user customer1 group1 v1`	Configures a new user to an SNMP group.

	Command or Action	Purpose
Step 4	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *read-view*] [**write** *write-view*] [**notify** *notify-view*] [**access** [**ipv6** *named-access-list*] [*acl-number*| *acl-name*]] **Example:** `Router(config)# snmp-server group group1 v1 context context1 read view1 write view1 notify view1`	Configures a new SNMP group or a table that maps SNMP users to SNMP views. • Use the **context** *context-name* keyword argument pair to associate the specified SNMP group with a configured SNMP context.
Step 5	**snmp-server view** *view-name oid-tree* {**included** | **excluded**} **Example:** `Router(config)# snmp-server view view1 ipForward included`	Creates or updates a view entry.
Step 6	**snmp-server enable traps** [*notification-type*] [**vrrp**] **Example:** `Router(config)# snmp-server enable traps`	Enables all SNMP notifications (traps or informs) available on your system.
Step 7	**snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number* | *extended-access-list-number* | *access-list-name*] **Example:** `Router(config)# snmp-server community public view view1 rw`	Sets up the community access string to permit access to the SNMP.
Step 8	**snmp-server host** {*hostname* | *ip-address*} [**vrf** *vrf-name*] [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] **Example:** `Router(config)# snmp-server host 10.0.0.1 vrf vrf1 public udp-port 7002`	Specifies the recipient of an SNMP notification operation.
Step 9	**snmp mib community-map** *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*][**target-list** *upn-list-name*] **Example:** `Router(config)# snmp mib community-map community1 context context1 target-list commAVpn`	Associates an SNMP community with an SNMP context, Engine ID, or security name.

	Command or Action	Purpose
Step 10	**snmp mib target list** *vpn-list-name* {**vrf** *vrf-name* | **host** *ip-address*} **Example:** `Router(config)# snmp mib target list commAVpn vrf vrf1`	Creates a list of target VRFs and hosts to associate with an SNMP community.
Step 11	**no snmp-server trap authentication vrf** **Example:** `Router(config)# no snmp-server trap authentication vrf`	(Optional) Disables all SNMP authentication notifications (traps and informs) generated for packets received on VRF interfaces. • Use this command to disable authentication traps only for those packets on VRF interfaces with incorrect community associations.

Configuration Examples for SNMP Support over VPNs--Context-Based Access Control

Example Configuring Context-Based Access Control

The following configuration example shows how to configure the SNMP Support over VPNs--Context-Based Access Control feature for SNMPv3:

Note In Cisco IOS Release 15.0(1)M and later releases, the **context** command is replaced by the **snmp context** command. See the Network Management Command Reference for more information.

```
snmp-server context A
snmp-server context B
ip vrf CustomerA
 rd 100:110
 context A
 route-target export 100:1000
 route-target import 100:1000
!
ip vrf CustomerB
 rd 100:120
 context B
 route-target export 100:2000
 route-target import 100:2000
!
interface Ethernet3/1
 description Belongs to VPN A
 ip vrf forwarding CustomerA
 ip address 192.168.2.1 255.255.255.0

interface Ethernet3/2
 description Belongs to VPN B
```

```
 ip vrf forwarding CustomerB
 ip address 192.168.2.2 255.255.255.0

snmp-server user CustomerAv3authusr CustomerAv3grpauth v3 auth md5 passwdA
snmp-server user CustomerBv3authusr CustomerBv3grpauth v3 auth md5 passwdB
snmp-server group CustomerAv3grpauth v3 auth context A read CustomerAv3view write
CustomerAv3view notify CustomerAv3view
snmp-server group CustomerBv3grpauth v3 auth context B read CustomerBv3view write
CustomerBv3view notify CustomerBv3view
snmp-server view view1 internet included
snmp-server view view1 internet.6.3.16 included
snmp-server view view1 internet.6.3.17 included
snmp-server view view1 internet.6.3.18 included
snmp-server view CustomerAv3view ipForward included
snmp-server view CustomerAv3view ciscoPingMIB included
snmp-server view CustomerBv3view ipForward included
snmp-server view CustomerBv3view ciscoPingMIB included
snmp-server community public view view1 rw
snmp-server enable traps
snmp-server host 192.168.2.3 vrf CustomerA version 3 auth CustomerAv3authusr udp-port
7002
snmp-server host 192.168.2.4 vrf CustomerB version 3 auth CustomerBv3authusr udp-port
7002
```

The following configuration example shows how to configure the SNMP Support over VPNs--Context-Based Access Control feature for SNMPv1 or SNMPv2:

Note In Cisco IOS Release 15.0(1)M and later releases, the **context** command is replaced by the **snmp context** command. See the Network Management Command Reference for more information.

```
snmp-server context A
snmp-server context B
ip vrf Customer_A
 rd 100:110
 context A
 route-target export 100:1000
 route-target import 100:1000
!
ip vrf Customer_B
 rd 100:120
 context B
 route-target export 100:2000
 route-target import 100:2000
!
interface Ethernet3/1
 description Belongs to VPN A
 ip vrf forwarding CustomerA
 ip address 192.168.2.1 255.255.255.0

interface Ethernet3/2
 description Belongs to VPN B
 ip vrf forwarding CustomerB
 ip address 192.168.2.2 255.255.255.0
snmp-server user commA grp1A v1
snmp-server user commA grp2A v2c
snmp-server user commB grp1B v1
snmp-server user commB grp2B v2c
snmp-server group grp1A v1 context A read viewA write viewA notify viewA
snmp-server group grp1B v1 context B read viewB write viewB notify viewB
snmp-server view viewA ipForward included
snmp-server view viewA ciscoPingMIB included
snmp-server view viewB ipForward included
snmp-server view viewB ciscoPingMIB included
snmp-server enable traps
snmp-server host 192.168.2.3 vrf CustomerA commA udp-port 7002
snmp-server host 192.168.2.4 vrf CustomerB commB udp-port 7002
snmp mib community-map commA context A target-list commAvpn
! Configures source address validation
snmp mib community-map commB context B target-list commBvpn
```

```
! Configures source address validation
snmp mib target list commAvpn vrf CustomerA
! Configures a list of VRFs or from which community commA is valid
snmp mib target list commBvpn vrf CustomerB
! Configures a list of VRFs or from which community commB is valid
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Cisco IOS Network Management commands	*Cisco IOS Network Management Command Reference*
SNMP configuration	" Configuring SNMP Support " chapter in the *Cisco IOS Network Management Configuration Guide*
SNMP Support for VPNs	SNMP Notification Support for VPNs

Standards

Standard	Title
None	--

MIBs

MIB	MIBs Link
• CISCO-PING-MIB • IP-FORWARD-MIB • SNMP-VACM-MIB, *The View-based Access Control Model (ACM) MIB for SNMP*	To locate and download MIBs for selected platforms, Cisco software releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 1441	*Introduction to version 2 of the Internet-standard Network Management Framework*
RFC 1442	*Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1443	*Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2)*

RFC	Title
RFC 1444	*Conformance Statements for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1445	*Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1446	*Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1447	*Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1448	*Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1449	*Transport Mappings for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 1450	*Management Information Base for version 2 of the Simple Network Management Protocol (SNMPv2)*
RFC 2571	*An Architecture for Describing SNMP Management Frameworks*
RFC 2576	*Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework*

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for SNMP Support over VPNs--Context-Based Access Control

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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

Table 9 Feature Information for SNMP Support over VPNs--Context-Based Access Control

Feature Name	Releases	Feature Information
SNMP Support over VPNs--Context-Based Access Control	12.0(23)S 12.2(25)S 12.2(31)SB2 12.2(33)SRA 12.2(33)SXH 12.3(2)T 15.0(1)M 15.0(1)S Cisco IOS XE Release 3.1.0SG	The SNMP Support over VPNs--Context-Based Access Control feature provides the infrastructure for multiple SNMP context support in Cisco IOS software and VPN-aware MIB infrastructure using the multiple SNMP context support infrastructure.

Glossary

MPLS VPN --Multiprotocol Label Switching Virtual Private Network

NMS --Network Management System. System responsible for managing at least part of a network. An NMS is generally a reasonably powerful and well-equipped computer, such as an engineering workstation. NMSs communicate with agents to help keep track of network statistics and resources.

SNMP --Simple Network Management Protocol. Network management protocol used almost exclusively in TCP/IP networks. SNMP provides a means to monitor and control network devices and to manage configurations, statistics collection, performance, and security.

SNMP communities --Authentication scheme that enables an intelligent network device to validate SNMP requests.

SNMPv2c --Version 2c of the Simple Network Management Protocol. SNMPv2c supports centralized and distributed network management strategies and includes improvements in the Structure of Management Information (SMI), protocol operations, management architecture, and security.

SNMPv3 --Version 3 of the Simple Network Management Protocol. Interoperable standards-based protocol for network management. SNMPv3 provides secure access to devices by a combination of authenticating and encrypting packets over the network.

UDP --User Datagram Protocol. Connectionless transport layer protocol in the TCP/IP protocol stack. UDP is a simple protocol that exchanges datagrams without acknowledgments or guaranteed delivery, requiring that error processing and retransmission be handled by other protocols. UDP is defined in RFC 768.

VRF --A VPN routing and forwarding instance. A VRF consists of an IP routing table, a derived forwarding table, a set of interfaces that use the forwarding table, and a set of rules and routing protocols that determine what goes into the forwarding table. In general, a VRF includes the routing information that defines a customer VPN site that is attached to a PE router.

AES and 3-DES Encryption Support for SNMP Version 3

The AES and 3-DES Encryption Support for SNMP Version 3AES and 3-DES Encryption Support for SNMP Version 3 feature enhances the encryption capabilities of SNMP version 3. Data Encryption Standard (DES) support was introduced in Cisco IOS Release 12.0 and expanded in Cisco IOS Release 12.1. This support for Simple Network Management Protocol (SNMP) version 3 User-Based Security Model (USM) is complaint with RFC 3414, which defines DES as the only required method of message encryption for SNMP version 3 authPriv mode.

The AES and 3-DES Encryption Support for SNMP Version 3 feature adds Advanced Encryption Standard (AES) 128-bit encryption in compliance with RFC 3826. RFC 3826 extensions have been included in the SNMP-USM-AES-MIB. In addition, Cisco-specific extensions to support Triple-Data Encryption Algorithm (3-DES) and AES 192-bit and 256-bit encryption have been added to the CISCO-SNMP-USM-MIB. Additional information can be found in the Internet-Draft titled *Extension to the User-Based Security Model (USM) to Support Triple-DES EDE in "Outside" CBC Mode*, which can be found at the following URL: http://www.snmp.com/eso/draft-reeder-snmpv3-usm-3desede-00.txt.

Finding Feature Information

Your software release may not support all the features documented in this module. For the latest feature information and caveats, see 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 document.

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

Prerequisites for AES and 3-DES Encryption Support for SNMP Version 3

- The network management station (NMS) must support SNMP version 3 to use this feature of the SNMP agent.
- This feature is available in only Cisco IOS software images where encryption algorithms are supported.

Information About AES and 3-DES Encryption Support for SNMP Version 3

SNMP Architecture

The architecture for describing Internet Management Frameworks contained in RFC 3411 describes the SNMP engine as composed of the following components:

- Dispatcher
- Message Processing Subsystem
- Security Subsystem
- Access Control Subsystem

Applications make use of the services of these subsystems. It is important to understand the SNMP architecture and the terminology of the architecture to understand where the Security Model fits into the architecture and interacts with the other subsystems within the architecture. The information is contained in RFC 3411 and your are encouraged to review this RFC to obtain an understanding of the SNMP architecture and subsystem interactions.

Encryption Key Support

In the AES and 3-DES Encryption Support for SNMP Version 3 feature the Cipher Block Chaining/Data Encryption Standard (CBC-DES) is the privacy protocol. Originally only DES was supported (as per RFC 3414). This feature adds support for AES-128 (as per RFC 3826) and AES-192, AES-256 and 3-DES (as per CISCO-SNMP-USM-OIDS-MIB).

- AES encryption uses the Cipher Feedback (CFB) mode with encryption key sizes of 128, 192, or 256 bits.
- 3DES encryption uses the 168-bit key size for encryption.

The AES Cipher Algorithm in the SNMP User-based Security Model draft describes the use of AES with 128-bit key size. However, the other options are also implemented with the extension to use the USM. There is currently no standard for generating localized keys for 192- or 256-bit size keys for AES or for 168-bit size key for 3-DES. There is no authentication protocol available with longer keys.

Management Information Base Support

The AES and 3-DES Encryption Support for SNMP Version 3 feature supports the selection of privacy protocols through the CLI and the Management Information Base (MIB). A new standard MIB, SNMP-USM-AES-MIB, provides support for the 128-bit key in AES. The extended options of AES with 192- or 256-bit keys and 3-DES are supported as extensions to the SNMP-USM-MIB, in the Cisco-specific MIB, CISCO-SNMP-USM-EXT-MIB.

How to Configure AES and 3-DES Encryption Support for SNMP Version 3

Adding a New User to an SNMP Group

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*][**vrf** *vrf-name*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes {128** | **192** | **256}**} *privpassword*] {*acl-number* | *acl-name*}]

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enters privileged EXEC mode. • Enter your password when prompted.
Step 2	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.

Command or Action	Purpose
Step 3 **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*][**vrf** *vrf-name*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** [**ipv6** *nacl*] [**priv** {**des** | **3des** | **aes {128 | 192 |256}**} *privpassword*] {*acl-number* | *acl-name*}] **Example:** `Router(config)# snmp-server user new-user new-group v3 auth md5 secureone priv aes 128 privatetwo 2`	Adds an SNMP user, specifies a group to which the user belongs, specifies the authorization algorithm to be used (MD5 or SHA), specifies the privacy algorithm to be used (DES, 3-DES, AES, AES-192, or AES-256), and specifies the password to be associated with this privacy protocol.

Verifying SNMP User Configuration

To display information about the configured characteristics of Simple Network Management Protocol (SNMP) users, use the **show snmp user** command in privileged EXEC mode.

Note The **show snmp user** command displays all the users configured on the router. However, unlike other SNMP configurations, the **snmp-server user** command will not appear on the "show running" output.

SUMMARY STEPS

1. **enable**
2. **show snmp user** [*username*

DETAILED STEPS

Step 1 **enable**
Enters privileged EXEC mode. Enter your password when prompted.

Step 2 **show snmp user** [*username*
The following example specifies the username as abcd, the engine ID string as 00000090200000000C025808, and the storage type as nonvolatile:

Example:

```
Router# show snmp user
 abcd
User name: abcd
Engine ID: 00000090200000000C025808
storage-type: nonvolatile       active access-list: 10
Rowstatus: active
Authentication Protocol: MD5
Privacy protocol: 3DES
Group name: VacmGroupName
Group name: VacmGroupName
```

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
SNMP commands	*Cisco IOS Network Management Command Reference*
SNMP configuration tasks	*Cisco IOS Network Management Configuration Guide*

Standards

Standard	Title
draft-reeder-snmpv3-usm-3desede-00.txt	Extension to the User-Based Security Model (USM) to Support Triple-DES EDE in "Outside" CBC Mode

MIBs

MIB	MIBs Link
• CISCO-SNMP-USM-OIDS-MIB • SNMP-USM-AES-MIB	To locate and download MIBs for selected platforms, Cisco IOS releases, and feature sets, use Cisco MIB Locator found at the following URL: http://www.cisco.com/go/mibs

RFCs

RFC	Title
RFC 2574	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)
RFC 3411	Architecture for Describing Internet Management Frameworks
RFC 3414	*User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)*
RFC 3826	The Advanced Encryption Standard (AES) Cipher Algorithm in the SNMP User-based Security Model

Technical Assistance

Description	Link
The Cisco Support and Documentation website provides online resources to download documentation, software, and tools. Use these resources to install and configure the software and to troubleshoot and resolve technical issues with Cisco products and technologies. Access to most tools on the Cisco Support and Documentation website requires a Cisco.com user ID and password.	http://www.cisco.com/cisco/web/support/index.html

Feature Information for AES and 3-DES Encryption Support for SNMP Version 3

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

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

Table 10 Feature Information for AES and 3-DES Encryption Support for SNMP Version 3

Feature Name	Releases	Feature Information
AES and 3-DES Encryption Support for SNMP Version 3	12.2(33)SRB 12.2(33)SB 12.2(33)SXI 12.4(2)T 15.0(1)S Cisco IOS XE 3.1.0SG	The AES and 3-DES Encryption Support for SNMP Version 3 feature enhances the encryption capabilities of SNMP version 3. DES support was introduced in Cisco IOS Release 12.0 and expanded in Cisco IOS Release 12.1. This support for SNMP version 3 USM is complaint with RFC 3414, which defines DES as the only required method of message encryption for SNMP version 3 authPriv mode. The AES and 3-DES Encryption Support for SNMP Version 3 feature adds AES 128-bit encryption in compliance with RFC 3826. AES and 3-DES Encryption Support for SNMP Version 3 was introduced in Cisco IOS Release 12.4(2)T. AES and 3-DES Encryption Support for SNMP Version 3 was integrated into Cisco IOS Release 12.2(33)SRB. AES and 3-DES Encryption Support for SNMP Version 3 was integrated into Cisco IOS Release 12.2(33)SB. This feature was integrated into Cisco IOS Release 15.0(1)S.