MPLS Label Distribution Protocol MIB Version 8 Upgrade

First Published: November 13, 2000
Last Updated: June 29, 2007

The MPLS Label Distribution Protocol (LDP) MIB Version 8 Upgrade feature enhances the LDP MIB to support the Internet Engineering Task Force (IETF) draft Version 8.

History for MLPS Label Distribution Protocol MIB Version 8 Update Feature

Release	Modification
12.0(11)ST	This feature was introduced to provide SNMP agent support for the MPLS LDP MIB on Cisco 7200, Cisco 7500, and Cisco 12000 series routers.
12.2(2)T	This feature was added to this release to provide SNMP agent support for the MPLS LDP MIB on Cisco 7200 and Cisco 7500 series routers.
12.0(21)ST	This feature was added to this release to provide SNMP agent and LDP notification support for the MPLS LDP MIB on Cisco 7200, Cisco 7500, and Cisco 12000 series Internet routers.
12.0(22)S	This feature (Version 1) was integrated into Cisco IOS Release 12.0(22)S.
12.0(24)S	This feature was upgraded to Version 8 in Cisco IOS Release 12.0(24)S.
12.0(27)S	Support for the MPLS VPN—VPN Aware LDP MIB feature was added.
12.2(18)S	This feature was integrated into Cisco IOS Release 12.2(18)S.
12.2(33)SRA	This feature was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This feature was integrated into Cisco IOS Release 12.2(33)SXH.

Finding Support Information for Platforms and Cisco IOS and Catalyst OS Software Images

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

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Contents

Prerequisites for MPLS LDP MIB Version 8 Upgrade

- Simple Network Management Protocol (SNMP) must be installed and enabled on the label switch routers (LSRs).
- Multiprotocol Label Switching (MPLS) must be enabled on the LSRs.
- LDP must be enabled on the LSRs.

Restrictions for MPLS LDP MIB Version 8 Upgrade

This implementation of the MPLS LDP MIB is limited to read-only (RO) permission for MIB objects, except for MIB object mplsLdpSessionUpDownTrapEnable, which has been extended to be writable by the SNMP agent.

Setting this object to a value of true enables both the mplsLdpSessionUp and mplsLdpSessionDown notifications on the LSR; conversely, setting this object to a value of false disables both of these notifications.

For a description of notification events, see the "Events Generating MPLS LDP MIB Notifications in MPLS LDP MIB Version 8 Upgrade" section on page 9.

Most MPLS LDP MIB objects are set up automatically during the LDP peer discovery (hello) process and the subsequent negotiation of parameters and establishment of LDP sessions between the LDP peers.

The following tables are not implemented in this feature:

- mplsLdpEntityFrParmsTable
- mplsLdpEntityConfFrLRTable
- mplsLdpFrameRelaySesTable
- mplsFecTable
- mplsLdpSesInLabelMapTable

- mplsXCsFecsTable
- mplsLdpSesPeerAddrTable

Information About MPLS LDP MIB Version 8 Upgrade

To configure MPLS LDP MIB Version 8 Upgrade, you need to understand the following concepts:

- Feature Design of MPLS LDP MIB Version 8 Upgrade, page 3
- Enhancements in Version 8 of the MPLS LDP MIB, page 4
- Benefits of MPLS LDP MIB Version 8 Upgrade, page 5

Feature Design of MPLS LDP MIB Version 8 Upgrade

MPLS is a packet forwarding technology that uses a short, fixed-length value called a label in packets to specify the next hop for packet transport through an MPLS network by means of label switch routers (LSRs).

A fundamental MPLS principle is that LSRs in an MPLS network must agree on the definition of the labels being used for packet forwarding operations. Label agreement is achieved in an MPLS network by means of procedures defined in the LDP.

LDP operations begin with a discovery (hello) process, during which an LDP entity (a local LSR) finds a cooperating LDP peer in the network, and the two negotiate basic operating procedures. The recognition and identification of a peer by means of this discovery process results in a hello adjacency, which represents the context within which label binding information is exchanged between the local LSR and its LDP peer. LDP then creates an active LDP session between the two LSRs to effect the exchange of label binding information. When this process is carried to completion with respect to all of the LSRs in an MPLS network, the result is a label-switched path (LSP), which constitutes an end-to-end packet transmission pathway between the communicating network devices.

By means of LDP, LSRs can collect, distribute, and release label binding information to other LSRs in an MPLS network, thereby enabling the hop-by-hop forwarding of packets in the network along normally routed paths.

The MPLS LDP MIB has been implemented to enable standard, SNMP-based network management of the label switching features in Cisco IOS software. Providing this capability requires SNMP agent code to execute on a designated network management station (NMS) in the network. The NMS serves as the medium for user interaction with the network management objects in the MPLS LDP MIB.

The SNMP agent code has a layered structure that is compatible with Cisco IOS software and presents a network administrative and management interface to the objects in the MPLS LDP MIB and, thence, to the rich set of label switching capabilities supported by Cisco IOS software.

By means of an SNMP agent, you can access MPLS LDP MIB objects using standard SNMP GET operations, and you can use those objects to accomplish a variety of network management tasks. All the objects in the MPLS LDP MIB follow the conventions defined in the IETF draft MIB entitled *draft-ietf-mpls-ldp-mib-08.txt,* which defines network management objects in a structured and standardized manner. This draft MIB is evolving and is soon expected to be a standard. Accordingly, the MPLS LDP MIB will be implemented in such a way that it tracks the evolution of this IETF document.

However, slight differences exist between the IETF draft MIB and the implementation of equivalent Cisco IOS functions. As a result, some minor translations between the MPLS LDP MIB objects and the internal Cisco IOS data structures are needed. Such translations are accomplished by the SNMP agent, which runs in the background on the NMS workstation as a low-priority process.

The extensive Cisco IOS label switching capabilities provide an integrated approach to managing the large volumes of traffic carried by WANs. These capabilities are integrated into the Layer 3 network services, thus optimizing the routing of high-volume traffic through Internet service provider backbones while, at the same time, ensuring the resistance of the network to link or node failures.

Cisco IOS Release 12.0(11)ST and later releases support the following MPLS LDP MIB-related functions:

- Tag Distribution Protocol (TDP)
- Generation and sending of event notification messages that signal changes in the status of LDP sessions
- Enabling and disabling of event notification messages by means of extensions to existing SNMP CLI commands
- Specification of the name or the IP address of an NMS workstation in the operating environment to which Cisco IOS event notification messages are to be sent to serve network administrative and management purposes
- Storage of the configuration pertaining to an event notification message in NVRAM of the NMS

The structure of the MPLS LDP MIB conforms to Abstract Syntax Notation One (ASN.1), so the MIB forms a highly structured and idealized database of network management objects.

Using any standard SNMP application, you can retrieve and display information from the MPLS LDP MIB by means of standard SNMP GET and GETNEXT operations.

Note Because the MPLS LDP MIB was not given an Internet Assigned Numbers Authority (IANA) experimental object identifier (OID) at the time of its implementation, Cisco chose to implement the MIB under the ciscoExperimental OID number, as follows:

ciscoExperimental
1.3.6.1.4.1.9.10
mplsLdpMIB
1.3.6.1.4.1.9.10.65

If the MPLS LDP MIB is assigned an IANA Experimental OID number, Cisco will replace all objects in the MIB under the ciscoExperimental OID and reposition the objects under the IANA Experimental OID.

Enhancements in Version 8 of the MPLS LDP MIB

Version 8 of the MPLS LDP MIB contains the following enhancements:

- TDP support
- Upgraded objects
- New indexing that is no longer based on the number of sessions
- Multiple SNMP context support for Virtual Private Networks (VPNs)

Benefits of MPLS LDP MIB Version 8 Upgrade

- Supports TDP and LDP
- Establishes LDP sessions between peer devices in an MPLS network
- Retrieves MIB parameters relating to the operation of LDP entities, such as:
  - Well-known LDP discovery port
  - Maximum transmission unit (MTU)
  - Proposed keepalive timer interval
  - Loop detection
  - Session establishment thresholds
  - Range of virtual path identifier/virtual channel identifier (VPI/VCI) pairs to be used in forming labels
- Gathers statistics related to LDP operations, such as error counters (Table 5)
- Monitors the time remaining for hello adjacencies
- Monitors the characteristics and status of LDP peers, such as:
  - Internetwork layer address of LDP peers
  - Loop detection of the LDP peers
  - Default MTU of the LDP peer
  - Number of seconds the LDP peer proposes as the value of the keepalive interval
- Monitors the characteristics and status of LDP sessions, such as:
  - Displaying the error counters (Table 10)
  - Determining the LDP version being used by the LDP session
  - Determining the keepalive hold time remaining for an LDP session
  - Determining the state of an LDP session (whether the session is active or not)
  - Displaying the label ranges (Table 2) for platform-wide and interface-specific sessions
  - Displaying the ATM parameters (Table 3)

Description of MPLS LDP MIB Elements for MPLS LDP MIB Version 8 Upgrade

LDP operations related to an MPLS LDP MIB involve the following functional elements:

- LDP entity—Relates to an instance of LDP for purposes of exchanging label spaces; describes a potential session.
- LDP peer—Refers to a remote LDP entity (that is, a nonlocal LSR).
- LDP session—Refers to an active LDP process between a local LSR and a remote LDP peer.

- Hello adjacency—Refers to the result of an LDP discovery process that affirms the state of two LSRs in an MPLS network as being adjacent to each other (that is, as being LDP peers). When the neighbor is discovered, the neighbor becomes a hello adjacency. An LDP session can be established with the hello adjacency. After the session is established, label bindings can be exchanged between the LSRs.

These MPLS LDP MIB elements are briefly described under separate headings below.

In effect, the MPLS LDP MIB provides a network management database that supports real-time access to the various MIB objects in the database. This database reflects the current state of MPLS LDP operations in the network. You can access this network management information database by means of standard SNMP commands issued from an NMS in the MPLS LDP operating environment.

The MPLS LDP MIB supports the following network management and administrative activities:

- Retrieving MPLS LDP MIB parameters pertaining to LDP operations
- Monitoring the characteristics and the status of LDP peers
- Monitoring the status of LDP sessions between LDP peers
- Monitoring hello adjacencies in the network
- Gathering statistics regarding LDP sessions

LDP Entities

An LDP entity is uniquely identified by an LDP identifier that consists of the mplsLdpEntityLdpId and the mplsLdpEntityIndex (see Figure 1).

- The mplsLdpEntityLdpId consists of the local LSR ID (four octets) and the label space ID (two octets). The label space ID identifies a specific label space available within the LSR.
- The mplsLdpEntityIndex consists of the IP address of the peer active hello adjacency, which is the 32-bit representation of the IP address assigned to the peer LSR.

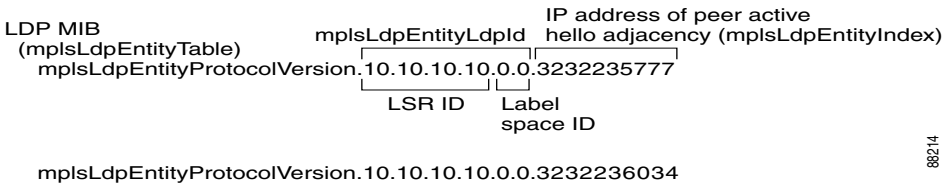
The mplsldpEntityProtocolVersion is a sample object from the mplsLdpEntityTable.

Figure 1 shows the following indexing:

- mplsLdpEntityLdpId = 10.10.10.10.0.0
- LSR ID = 10.10.10.10
- Label space ID = 0.0

  The mplsLdpEntityLdpId or the LDP ID consists of the LSR ID and the label space ID.
- The IP address of peer active hello adjacency or the mplsLdpEntityIndex = 3232235777, which is the 32-bit representation of the IP address assigned to the peer's active hello adjacency.

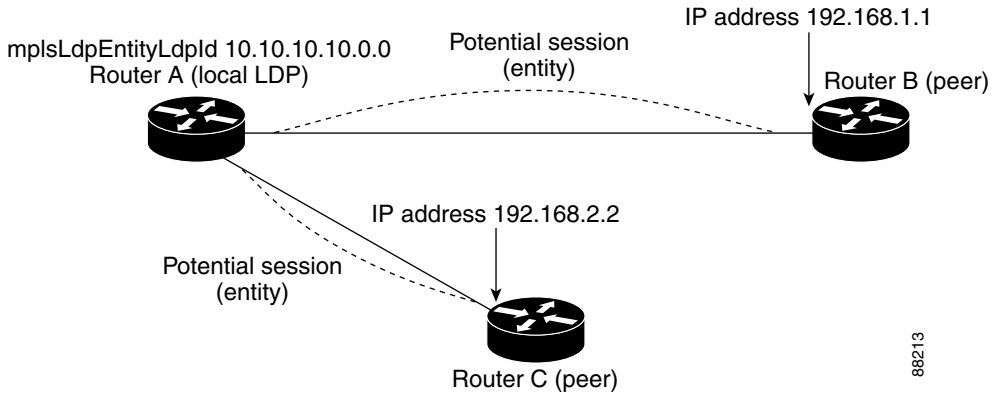
Figure 1 Sample Indexing for an LDP Entity



An LDP entity represents a label space that has the potential for a session with an LDP peer. An LDP entity is set up when a hello adjacency receives a hello message from an LDP peer.

In Figure 2, Router A has potential sessions with two remote peers, Routers B and C. The mplsLdpEntityLdpId is 10.10.10.10.0.0, and the IP address of the peer active hello adjacency (mplsLdpEntityIndex) is 3232235777, which is the 32-bit representation of the IP address 192.168.1.1 for Router B.

Figure 2 LDP Entity



LDP Sessions and Peers

LDP sessions exist between local entities and remote peers for the purpose of distributing label spaces. There is always a one-to-one correspondence between an LDP peer and an LDP session. A single LDP session is an LDP instance that communicates across one or more network links with a single LDP peer.

LDP supports the following types of sessions:

- Interface-specific—An interface-specific session uses interface resources for label space distributions. For example, each label-controlled ATM (LC-ATM) interface uses its own VPIs/VCIs for label space distributions. Depending on its configuration, an LDP platform can support zero, one, or more interface-specific sessions. Each LC-ATM interface has its own interface-specific label space and a nonzero label space ID.
- Platform-wide—An LDP platform supports a single platform-wide session for use by all interfaces that can share the same global label space. For Cisco platforms, all interface types except LC-ATM use the platform-wide session and have a label space ID of zero.

When a session is established between two peers, entries are created in the mplsLdpPeerTable and the mplsLdpSessionTable because they have the same indexing.

In Figure 3, Router A has two remote peers, Routers B and C. Router A has a single platform-wide session that consists of two serial interfaces with Router B and another platform-wide session with Router C. Router A also has two interface-specific sessions with Router B.

Figure 3 LDP Sessions

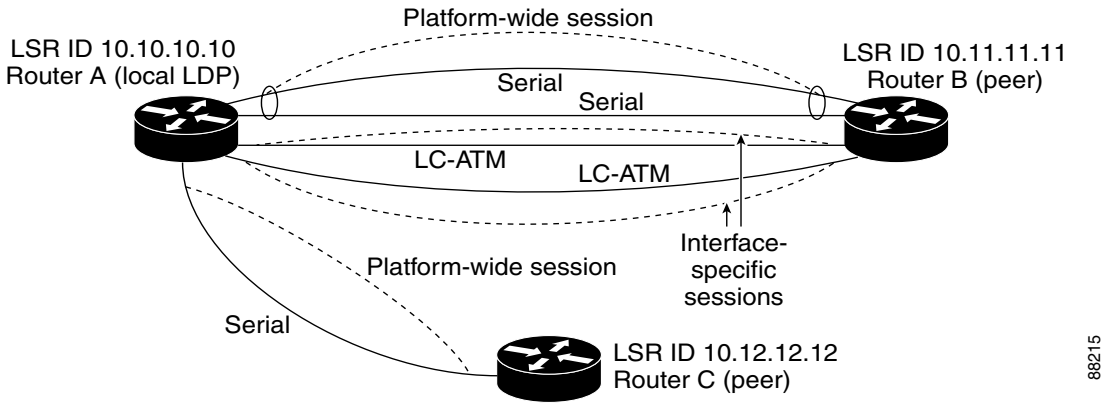


Figure 4 shows entries that correspond to the mplsLdpPeerTable and the mplsLdpSessionTable in Figure 3.

In Figure 4, mplsLdpSesState is a sample object from the mplsLdpSessionTable on Router A. There are four mplsLdpSesState sample objects shown (top to bottom). The first object represents a platform-wide session associated with two serial interfaces. The next two objects represent interface-specific sessions for the LC-ATM interfaces on Routers A and B. These interface-specific sessions have nonzero peer label space IDs. The last object represents a platform-wide session for the next peer, Router C.

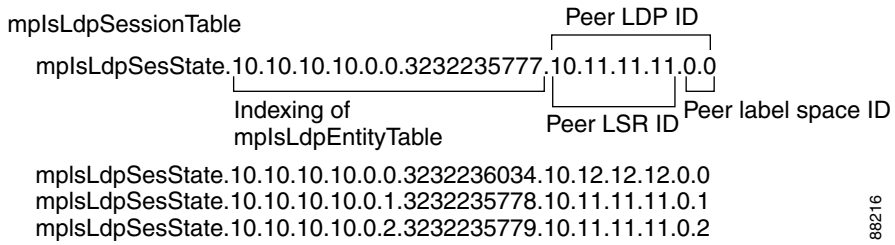
The indexing is based on the entries in the mplsLdpEntityTable. It begins with the indexes of the mplsLdpEntityTable and adds the following:

- Peer LDP ID = 10.11.11.11.0.0

  The peer LDP ID consists of the peer LSR ID (four octets) and the peer label space ID (two octets).
- Peer LSR ID = 10.11.11.11
- Peer label space ID = 0.0

  The peer label space ID identifies a specific peer label space available within the LSR.

Figure 4 Sample Indexing for an LDP Session



LDP Hello Adjacencies

An LDP hello adjacency is a network link between a router and its peers. An LDP hello adjacency enables two adjacent peers to exchange label binding information.

An LDP hello adjacency exists for each link on which LDP runs. Multiple LDP hello adjacencies exist whenever there is more than one link in a session between a router and its peer, such as in a platform-wide session.

A hello adjacency is considered active if it is currently engaged in a session, or nonactive if it is not currently engaged in a session.

A targeted hello adjacency is not directly connected to its peer and has an unlimited number of hops between itself and its peer. A linked hello adjacency is directly connected between two routers.

In Figure 5, Router A has two remote peers, Routers B and C. Router A has a platform-wide session with Router B that consists of three serial interfaces, one of which is active and another platform-wide (targeted) session with Router C.

Figure 5 Hello Adjacency

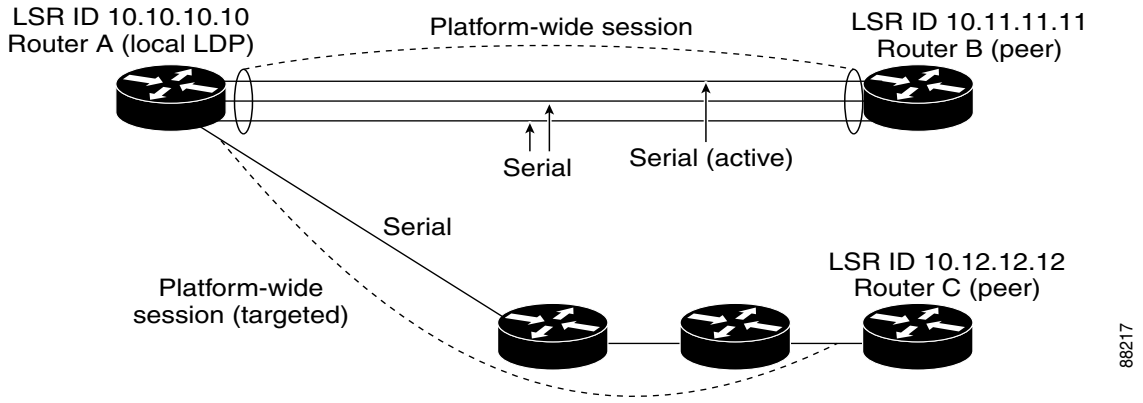
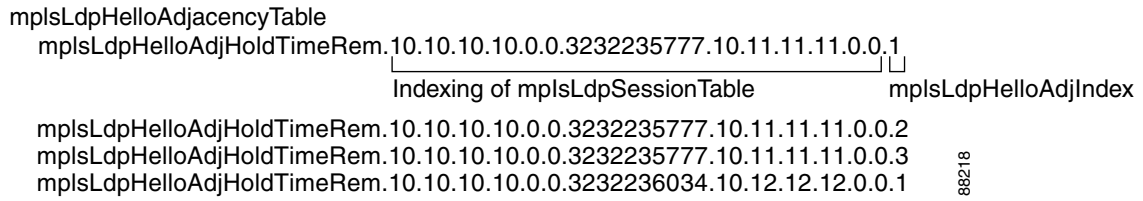


Figure 6 shows entries in the mplsLdpHelloAdjacencyTable. There are four mplsLdpHelloAdjHoldTime sample objects (top to bottom). They represent the two platform-wide sessions and the four serial links shown in Figure 5.

The indexing is based on the mplsLdpSessionTable. When the mplsLdpHelloAdjIndex enumerates the different links within a single session, the active link is mplsLdpHelloAdjIndex = 1.

Figure 6 Sample Indexing for an LDP Hello Adjacency



Events Generating MPLS LDP MIB Notifications in MPLS LDP MIB Version 8 Upgrade

When you enable MPLS LDP MIB notification functionality by issuing the **snmp-server enable traps mpls ldp** command, notification messages are generated and sent to a designated NMS in the network to signal the occurrence of specific events within Cisco IOS.

The MPLS LDP MIB objects involved in LDP status transitions and event notifications include the following:

- mplsLdpSessionUp—This message is generated when an LDP entity (a local LSR) establishes an LDP session with another LDP entity (an adjacent LDP peer in the network).

- mplsLdpSessionDown—This message is generated when an LDP session between a local LSR and its adjacent LDP peer is terminated.
- mplsLdpPathVectorLimitMismatch—This message is generated when a local LSR establishes an LDP session with its adjacent peer LSR, but the two LSRs have dissimilar path vector limits.

  The value of the path vector limit can range from 0 through 255; a value of 0 indicates that loop detection is off; any value other than zero up to 255 indicates that loop detection is on and, in addition, specifies the maximum number of hops through which an LDP message can pass before a loop condition in the network is sensed.

  We recommend that all LDP-enabled routers in the network be configured with the same path vector limit. Accordingly, the mplsLdpPathVectorLimitMismatch object exists in the MPLS LDP MIB to provide a warning message to the NMS when two routers engaged in LDP operations have different path vector limits.

Note This notification is generated only if the distribution method is downstream-on-demand.

- mplsLdpFailedInitSessionThresholdExceeded—This message is generated when a local LSR and an adjacent LDP peer attempt to set up an LDP session between them, but fail to do so after a specified number of attempts. The default number of attempts is 8. This default value is implemented and cannot be changed.

  Eight failed attempts to establish an LDP session between a local LSR and an LDP peer, due to any type of incompatibility between the devices, causes this notification message to be generated. Cisco routers support the same features across multiple platforms.

  Therefore, the most likely incompatibility to occur between Cisco LSRs is a mismatch of their respective ATM VPI/VCI label ranges.

  For example, if you specify a range of valid labels for an LSR that does not overlap the range of its adjacent LDP peer, the routers try eight times to create an LDP session between themselves before the mplsLdpFailedInitSessionThresholdExceeded notification is generated and sent to the NMS as an informational message.

  The LSRs whose label ranges do not overlap continue their attempt to create an LDP session between themselves after the eight-retry threshold is exceeded.

  In such cases, the LDP threshold exceeded notification alerts the network administrator about a condition in the network that might warrant attention.

  RFC 3036, *LDP Specification*, details the incompatibilities that can exist between Cisco routers and/or other vendor LSRs in an MPLS network.

  Among such incompatibilities, for example, are the following:

  - Nonoverlapping ATM VPI/VCI ranges (as noted above) or nonoverlapping Frame-Relay DLCI ranges between LSRs attempting to set up an LDP session
  - Unsupported label distribution method
  - Dissimilar protocol data unit (PDU) sizes
  - Dissimilar types of LDP feature support

MIB Tables in MPLS LDP MIB Version 8 Upgrade

Version 8 of the MPLS LDP MIB consists of the following tables:

- mplsLdpEntityTable (Table 1)—Contains entries for every active LDP hello adjacency. Nonactive hello adjacencies appear in the mplsLdpHelloAdjacencyTable, rather than this table. This table is indexed by the local LDP identifier for the interface and the IP address of the peer active hello adjacency. (See Figure 1.)

  The advantage of showing the active hello adjacency instead of sessions in this table is that the active hello adjacency can exist even if an LDP session is not active (cannot be established). Previous implementations of the IETF MPLS-LDP MIB used sessions as the entries in this table. This approach was inadequate because as sessions went down, the entries in the entity table would disappear completely because the agent code could no longer access them. This resulted in the MIB failing to provide information about failed LDP sessions.

  Directed adjacencies are also shown in this table. These entries, however, are always up administratively (adminStatus) and operationally (operStatus), because the adjacencies disappear if the directed session fails. Nondirected adjacencies might disappear from the MIB on some occasions, because adjacencies are deleted if the underlying interface becomes operationally down, for example.

- mplsLdpEntityConfGenLRTable (Table 2)—Contains entries for every LDP-enabled interface that is in the global label space. (For Cisco, this applies to all interfaces except LC-ATM. LC-ATM entities are shown in the mplsLdpEntityConfAtmLRTable instead.) Indexing is the same as it is for the mplsLdpEntityTable, except two indexes have been added, mplsLdpEntityConfGenLRMin and mplsLdpEntityConfGenLRMax. These additional indexes allow more than one label range to be defined. However, in the current Cisco IOS implementation, only one global label range is allowed.
- mplsLdpEntityAtmParmsTable (Table 3)—Contains entries for every LDP-enabled LC-ATM interface. This table is indexed the same as the mplsLdpEntityTable although only LC-ATM interfaces are shown.
- mplsLdpEntityConfAtmLRTable (Table 4)—Contains entries for every LDP-enabled LC-ATM interface. Indexing is the same as it is for the mplsLdpEntityTable, except two indexes have been added, mplsLdpEntityConfAtmLRMinVpi and mplsLdpEntityConfAtmLRMinVci. These additional indexes allow more than one label range to be defined. However, in the current Cisco IOS implementation, only one label range per LC-ATM interface is allowed.
- mplsLdpEntityStatsTable (Table 5)—Augments the mplsLdpEntityTable and shares the exact same indexing for performing GET and GETNEXT operations. This table shows additional statistics for entities.
- mplsLdpPeerTable (Table 6)—Contains entries for all peer sessions. This table is indexed by the local LDP identifier of the session, the IP address of the peer active hello adjacency, and the peer's LDP identifier. (See Figure 4.)
- mplsLdpHelloAdjacencyTable (Table 7)—Contains entries for all hello adjacencies. This table is indexed by the local LDP identifier of the associated session, the IP address of the peer active hello adjacency, the LDP identifier for the peer, and an arbitrary index that is set to the list position of the adjacency. (See Figure 6.)
- mplsLdpSessionTable (Table 8)—Augments the mplsLdpPeerTable and shares the same indexing for performing GET and GETNEXT operations. This table shows all sessions.

- mplsLdpAtmSesTable (Table 9)—Contains entries for LC-ATM sessions. Indexing is the same as it is for the mplsLdpPeerTable, except two indexes have been added, mplsLdpSesAtmLRLowerBoundVpi and mplsLdpSesAtmLRLowerBoundVci. These additional indexes allow more than one label range to be defined. However, in the current Cisco IOS implementation, only one label range per LC-ATM interface is allowed.
- mplsLdpSesStatsTable (Table 10)—Augments the mplsLdpPeerTable and shares the exact same indexing for performing GET and GETNEXT operations. This table shows additional statistics for sessions.

mplsLdpEntityTable

Table 1 lists the mplsLdpEntityTable objects and their descriptions.

Table 1 mplsLdpEntityTable Objects and Descriptions

Object	Description
mplsLdpEntityEntry	Represents an LDP entity, which is a potential session between two peers.
mplsLdpEntityLdpId	The LDP identifier (not accessible) consists of the local LSR ID (four octets) and the label space ID (two octets).
mplsLdpEntityIndex	A secondary index that identifies this row uniquely. It consists of the IP address of the peer active hello adjacency, which is the 32-bit representation of the IP address assigned to the LSR (not accessible).
mplsLdpEntityProtocolVersion	The version number of the LDP protocol to be used in the session initialization message.
mplsLdpEntityAdminStatus	The administrative status of this LDP entity is always up. If the hello adjacency fails, this entity disappears from the mplsLdpEntityTable.
mplsLdpEntityOperStatus	The operational status of this LDP entity. Values are unknown(0), enabled(1), and disabled(2).
mplsLdpEntityTcpDscPort	The TCP discovery port for LDP or TDP. The default value is 646 (LDP).
mplsLdpEntityUdpDscPort	The UDP discovery port for LDP or TDP. The default value is 646 (LDP).
mplsLdpEntityMaxPduLength	The maximum PDU length that is sent in the common session parameters of an initialization message.
mplsLdpEntityKeepAliveHoldTimer	The two-octet value that is the proposed keepalive hold time for this LDP entity.
mplsLdpEntityHelloHoldTimer	The two-octet value that is the proposed hello hold time for this LDP entity.
mplsLdpEntityInitSesThreshold	The threshold for notification when this entity and its peer are engaged in an endless sequence of initialization messages. The default value is 8 and cannot be changed by SNMP or CLI.

Table 1 mplsLdpEntityTable Objects and Descriptions (continued)

Object	Description
mplsLdpEntityLabelDistMethod	The specified method of label distribution for any given LDP session. Values are downstreamOnDemand(1) and downstreamUnsolicited(2).
mplsLdpEntityLabelRetentionMode	Can be configured to use either conservative(1) for LC-ATM or liberal(2) for all other interfaces.
mplsLdpEntityPVLMisTrapEnable	Indicates whether the mplsLdpPVLMismatch trap should be generated. If the value is enabled(1), the trap is generated. If the value is disabled(2), the trap is not generated. The default is disabled(2). **Note** The mplsLdpPVLMismatch trap is generated only if mplsLdpEntityLabelDistMethod is downstreamOnDemand(1).
mplsLdpEntityPVL	If the value of this object is 0, loop detection for path vectors is disabled. Otherwise, if this object has a value greater than zero, loop detection for path vectors is enabled, and the path vector limit is this value. **Note** The mplsLdpEntityPVL object is non-zero only if mplsLdpEntityLabelDistMethod is downstreamOnDemand(1).
mplsLdpEntityHopCountLimit	If the value of this object is 0, loop detection using hop counters is disabled. If the value of this object is greater than 0, loop detection using hop counters is enabled, and this object specifies this entity's maximum allowable value for the hop count. **Note** The mplsLdpEntityHopCountLimit object is non-zero only if mplsLdpEntityLabelDistMethod is downstreamOnDemand(1).
mplsLdpEntityTargPeer	If this LDP entity uses a targeted adjacency, this object is set to true(1). The default value is false(2).
mplsLdpEntityTargPeerAddrType	The type of the internetwork layer address used for the extended discovery. This object indicates how the value of mplsLdpEntityTargPeerAddr is to be interpreted.
mplsLdpEntityTargPeerAddr	The value of the internetwork layer address used for the targeted adjacency.
mplsLdpEntityOptionalParameters	Specifies the optional parameters for the LDP initialization message. If the value is generic(1), no optional parameters are sent in the LDP initialization message associated with this entity. LC-ATM uses atmParameters(2) to specify that a row in the mplsLdpEntityAtmParmsTable corresponds to this entry. **Note** Frame Relay parameters are not supported.

Table 1 mplsLdpEntityTable Objects and Descriptions (continued)

Object	Description
mplsLdpEntityDiscontinuityTime	The value of sysUpTime on the most recent occasion when one or more of this entity's counters suffered a discontinuity. The relevant counters are the specific instances of any Counter32 or Counter64 object contained in the mplsLdpEntityStatsTable that are associated with this entity. If no such discontinuities have occurred since the last reinitialization of the local management subsystem, this object contains a 0 value.
mplsLdpEntityStorType	The storage type for this entry is a read-only implementation that is always volatile.
mplsLdpEntityRowStatus	This object is a read-only implementation that is always active.

mplsLdpEntityConfGenLRTable

Table 2 lists the mplsLdpEntityConfGenLRTable objects and their descriptions.

Table 2 mplsLdpEntityConfGenLRTable Objects and Descriptions

Object	Description
mplsLdpEntityConfGenLREntry	A row in the LDP Entity Configurable Generic Label Range table. One entry in this table contains information on a single range of labels; the range is defined by an upper boundary (VPI/VCI pair) and a lower boundary (VPI/VCI pair). The current implementation supports one label range per entity.
mplsLdpEntityConfGenLRMin	The minimum label configured for this range (not accessible).
mplsLdpEntityConfGenLRMax	The maximum label configured for this range (not accessible).
mplsLdpEntityConfGenIfIndxOrZero	This value represents the SNMP IF-MIB index for the platform-wide entity. If the active hello adjacency is targeted, the value is 0.
mplsLdpEntityConfGenLRStorType	The storage type for this entry is a read-only implementation that is always volatile.
mplsLdpEntityConfGenLRRowStatus	This object is a read-only implementation that is always active.

mplsLdpEntityAtmParmsTable

Table 3 lists the mplsLdpEntityAtmParmsTable objects and their descriptions.

Table 3 mplsLdpEntityAtmParmsTable Objects and Descriptions

Object	Description
mplsLdpEntityAtmParmsEntry	Represents the ATM parameters and ATM information for this LDP entity.
mplsLdpEntityAtmIfIndxOrZero	This value represents the SNMP IF-MIB index for the interface-specific LC-ATM entity.
mplsLdpEntityAtmMergeCap	Denotes the merge capability of this entity.
mplsLdpEntityAtmLRComponents	Number of label range components in the initialization message. This also represents the number of entries in the mplsLdpEntityConfAtmLRTable that correspond to this entry.
mplsLdpEntityAtmVcDirectionality	If the value of this object is bidirectional(0), a given VCI within a given VPI is used as a label for both directions independently of one another. If the value of this object is unidirectional(1), a given VCI within a VPI designates one direction.
mplsLdpEntityAtmLsrConnectivity	The peer LSR can be connected indirectly by means of an ATM VP, so that the VPI values can be different on the endpoints. For that reason, the label must be encoded entirely within the VCI field. Values are direct(1), the default, and indirect(2).
mplsLdpEntityDefaultControlVpi	The default VPI value for the non-MPLS connection.
mplsLdpEntityDefaultControlVci	The default VCI value for the non-MPLS connection.
mplsLdpEntityUnlabTrafVpi	VPI value of the VCC supporting unlabeled traffic. This non-MPLS connection is used to carry unlabeled (IP) packets.
mplsLdpEntityUnlabTrafVci	VCI value of the VCC supporting unlabeled traffic. This non-MPLS connection is used to carry unlabeled (IP) packets.
mplsLdpEntityAtmStorType	The storage type for this entry is a read-only implementation that is always volatile.
mplsLdpEntityAtmRowStatus	This object is a read-only implementation that is always active.

mplsLdpEntityConfAtmLRTable

Table 4 lists the mplsLdpEntityConfAtmLRTable objects and their descriptions.

Table 4 mplsLdpEntityConfAtmLRTable Objects and Descriptions

Object	Description
mplsLdpEntityConfAtmLREntry	A row in the LDP Entity Configurable ATM Label Range Table. One entry in this table contains information on a single range of labels; the range is defined by an upper boundary (VPI/VCI pair) and a lower boundary (VPI/VCI pair). This is the same data used in the initialization message. This label range should overlap the label range of the peer.
mplsLdpEntityConfAtmLRMinVpi	The minimum VPI number configured for this range (not accessible).
mplsLdpEntityConfAtmLRMinVci	The minimum VCI number configured for this range (not accessible).
mplsLdpEntityConfAtmLRMaxVpi	The maximum VPI number configured for this range (not accessible).
mplsLdpEntityConfAtmLRMaxVci	The maximum VCI number configured for this range (not accessible).
mplsLdpEntityConfAtmLRStorType	The storage type for this entry is a read-only implementation that is always volatile.
mplsLdpEntityConfAtmLRRowStatus	This object is a read-only implementation that is always active.

mplsLdpEntityStatsTable

Table 5 lists the mplsLdpEntityStatsTable objects and their descriptions.

Table 5 mplsLdpEntityStatsTable Objects and Descriptions

Object	Description
mplsLdpEntityStatsEntry	These entries augment the mplsLdpEntityTable by providing additional information for each entry.
mplsLdpAttemptedSessions	Not supported in this feature.
mplsLdpSesRejectedNoHelloErrors	A count of the session rejected/no hello error notification messages sent or received by this LDP entity.
mplsLdpSesRejectedAdErrors	A count of the session rejected/parameters advertisement mode error notification messages sent or received by this LDP entity.
mplsLdpSesRejectedMaxPduErrors	A count of the session rejected/parameters max PDU length error notification messages sent or received by this LDP entity.
mplsLdpSesRejectedLRErrors	A count of the session rejected/parameters label range notification messages sent or received by this LDP entity.
mplsLdpBadLdpIdentifierErrors	A count of the number of bad LDP identifier fatal errors detected by the session associated with this LDP entity.

Table 5 mplsLdpEntityStatsTable Objects and Descriptions (continued)

Object	Description
mplsLdpBadPduLengthErrors	A count of the number of bad PDU length fatal errors detected by the session associated with this LDP entity.
mplsLdpBadMessageLengthErrors	A count of the number of bad message length fatal errors detected by the session associated with this LDP entity.
mplsLdpBadTlvLengthErrors	A count of the number of bad Type-Length-Value (TLV) length fatal errors detected by the session associated with this LDP entity.
mplsLdpMalformedTlvValueErrors	A count of the number of malformed TLV value fatal errors detected by the session associated with this LDP entity.
mplsLdpKeepAliveTimerExpErrors	A count of the number of session keepalive timer expired errors detected by the session associated with this LDP entity.
mplsLdpShutdownNotifReceived	A count of the number of shutdown notifications received related to the session associated with this LDP entity.
mplsLdpShutdownNotifSent	A count of the number of shutdown notifications sent related to the session associated with this LDP entity.

mplsLdpPeerTable

Table 6 lists the mplsLdpPeerTable objects and their descriptions.

Table 6 mplsLdpPeerTable Objects and Descriptions

Object	Description
mplsLdpPeerEntry	Information about a single peer that is related to a session (not accessible). **Note** This table is augmented by the mplsLdpSessionTable.
mplsLdpPeerLdpId	The LDP identifier of this LDP peer (not accessible) consists of the peer LSR ID (four octets) and the peer label space ID (two octets).
mplsLdpPeerLabelDistMethod	For any given LDP session, the method of label distribution. Values are downstreamOnDemand(1) and downstreamUnsolicited(2).

Table 6 mplsLdpPeerTable Objects and Descriptions (continued)

Object	Description
mplsLdpPeerLoopDetectionForPV	An indication of whether loop detection based on path vectors is disabled or enabled for this peer. For downstream unsolicited distribution (mplsLdpPeerLabelDistMethod is downstreamUnsolicited(2)), this object always has a value of disabled(0) and loop detection is disabled. For downstream-on-demand distribution (mplsLdpPeerLabelDistMethod is downstreamOnDemand(1)), this object has a value of enabled(1), provided that loop detection based on path vectors is enabled.
mplsLdpPeerPVL	If the value of mplsLdpPeerLoopDetectionForPV for this entry is enabled(1), this object represents that path vector limit for this peer. If the value of mplsLdpPeerLoopDetectionForPV for this entry is disabled(0), this value should be 0.

mplsLdpHelloAdjacencyTable

Table 7 lists the mplsLdpHelloAdjacencyTable objects and their descriptions.

Table 7 mplsLdpHelloAdjacencyTable Objects and Descriptions

Object	Description
mplsLdpHelloAdjacencyEntry	Each row represents a single LDP hello adjacency. An LDP session can have one or more hello adjacencies (not accessible).
mplsLdpHelloAdjIndex	An identifier for this specific adjacency (not accessible). The active hello adjacency has mplsLdpHelloAdjIndex equal to 1.
mplsLdpHelloAdjHoldTimeRem	The time remaining for this hello adjacency. This interval changes when the next hello message, which corresponds to this hello adjacency, is received.
mplsLdpHelloAdjType	This adjacency is the result of a link hello if the value of this object is link(1). Otherwise, this adjacency is a result of a targeted hello and its value is targeted(2).

mplsLdpSessionTable

Table 8 lists the mplsLdpSessionTable objects and their descriptions.

Table 8 mplsLdpSessionTable Objects and Descriptions

Object	Description
mplsLdpSessionEntry	An entry in this table represents information on a single session between an LDP entity and an LDP peer. The information contained in a row is read-only. This table augments the mplsLdpPeerTable.
mplsLdpSesState	The current state of the session. All of the states are based on the LDP or TDP state machine for session negotiation behavior. The states are as follows: • nonexistent(1) • initialized(2) • openrec(3) • opensent(4) • operational(5)
mplsLdpSesProtocolVersion	The version of the LDP protocol which this session is using. This is the version of the LDP protocol that has been negotiated during session initialization.
mplsLdpSesKeepAliveHoldTimeRem	The keepalive hold time remaining for this session.
mplsLdpSesMaxPduLen	The value of maximum allowable length for LDP PDUs for this session. This value could have been negotiated during the session initialization.
mplsLdpSesDiscontinuityTime	The value of sysUpTime on the most recent occasion when one or more of this session's counters suffered a discontinuity. The relevant counters are the specific instances of any Counter32 or Counter64 object contained in the mplsLdpSesStatsTable associated with this session. The initial value of this object is the value of sysUpTime when the entry was created in this table.

mplsLdpAtmSesTable

Table 9 lists the mplsLdpAtmSesTable objects and their descriptions.

Table 9 mplsLdpAtmSesTable Objects and Descriptions

Objects	Description
mplsLdpAtmSesEntry	An entry in this table represents information on a single label range intersection between an LDP entity and an LDP peer (not accessible).
mplsLdpAtmSesLRLowerBoundVpi	The minimum VPI number for this range (not accessible).
mplsLdpAtmSesLRLowerBoundVci	The minimum VCI number for this range (not accessible).

Table 9 mplsLdpAtmSesTable Objects and Descriptions (continued)

Objects	Description
mplsLdpAtmSesLRUpperBoundVpi	The maximum VPI number for this range (read-only).
mplsLdpAtmSesLRUpperBoundVci	The maximum VCI number for this range (read-only).

mplsLdpSesStatsTable

Table 10 lists the mplsLdpSesStatsTable objects and their descriptions.

Table 10 mplsLdpSesStatsTable Objects and Descriptions

Object	Description
mplsLdpSesStatsEntry	An entry in this table represents statistical information on a single session between an LDP entity and an LDP peer. This table augments the mplsLdpPeerTable.
mplsLdpSesStatsUnkMesTypeErrors	This object is the count of the number of unknown message type errors detected during this session.
mplsLdpSesStatsUnkTlvErrors	This object is the count of the number of unknown TLV errors detected during this session.

VPN Contexts in MPLS LDP MIB Version 8 Upgrade

Within an MPLS Border Gateway Protocol (BGP) 4 Virtual Private Network (VPN) environment, separate LDP processes can be created for each VPN. These processes and their associated data are called LDP contexts. Each context is independent from all others and contains data specific only to that context.

Cisco IOS Release 12.0(11)ST and later releases include the VPN Aware LDP MIB feature that allows the LDP MIB to get VPN context information. The feature adds support for different contexts for different MPLS VPNs. Users of the MIB can view MPLS LDP processes for a given MPLS VPN. The VPN Aware LDP MIB feature does not change the syntax of the IETF MPLS-LDP MIB. It changes the number and types of entries within the tables.

The IETF MPLS-LDP MIB can show information about only one context at a time. You can specify a context, either a global context or an MPLS VPN context, using an SMNP security name.

The following sections describe topics related to the VPN Aware LDP MIB feature:

- SNMP Contexts, page 21
- VPN Aware LDP MIB Sessions, page 21
- VPN Aware LDP MIB Notifications, page 23

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 users of other VPNs on the same networking device.

VPN-aware SNMP requires that SNMP manager and agent entities operating in a VPN environment agree on mapping between the SNMP security name and the VPN name. This mapping is created by using different contexts for the SNMP data of different VPNs, which is accomplished through the configuration of the SNMP View-based Access Control Model MIB (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 within the context of only that VPN.

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

- The first security phase is authentication of the username. During this phase, the user is authorized for SNMP access.
- The second phase is access control. During this phase, the user is authorized for SNMP access to the group objects in the requested SNMP context.
- In the third phase, the user can access a particular instance of a table entry. With this third phase, complete retrieval can be based on the SNMP context name.

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

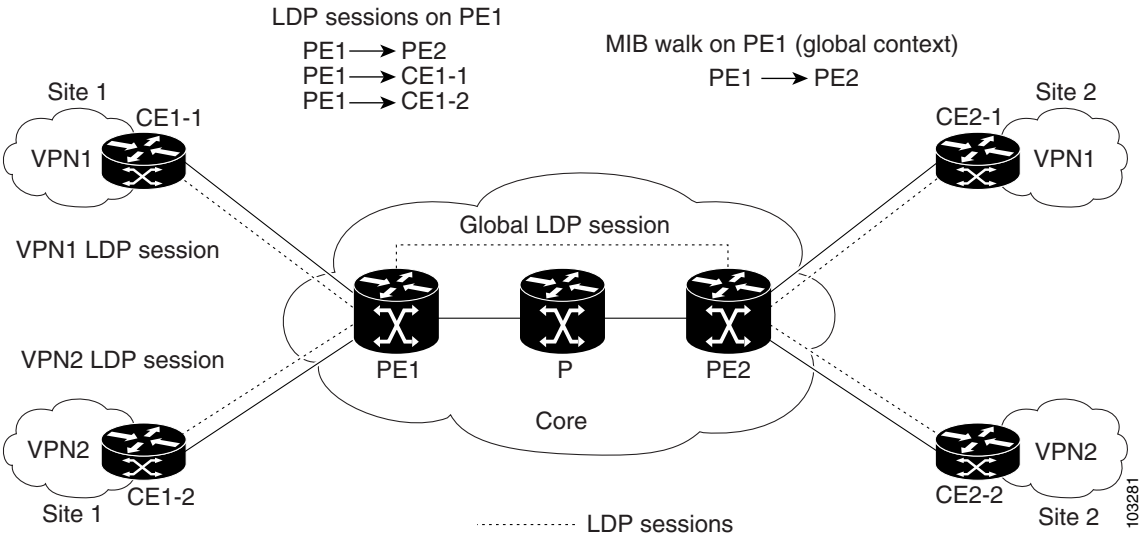
You can also enable or disable authentication traps for SNMP packets dropped due to VRF mismatches. By default, if SNMP authentication traps are enabled, VRF authentication traps are also enabled.

VPN Aware LDP MIB Sessions

Prior to Cisco IOS Release 12.0(11)ST, an SNMP query to the MPLS LDP MIB returned information about global sessions only. A query did not return information about LDP sessions in a VPN context. The IETF MPLS LDP MIB retrieved information from global routing tables, but did not retrieve information from VPN routing and forwarding instances (VRFs) that store per-VPN routing data. The MPLS LDP MIB looked only at LDP processes in the global context and ignored all other sessions. A query on a VRF returned no information. You can view LDP processes in a VPN context.

Figure 7 shows a sample MPLS VPN network with the MPLS LDP sessions prior to the implementation of the VPN Aware LDP MIB feature.

Figure 7 MPLS LDP Sessions Setup Before VPN Aware LDP MIB Feature



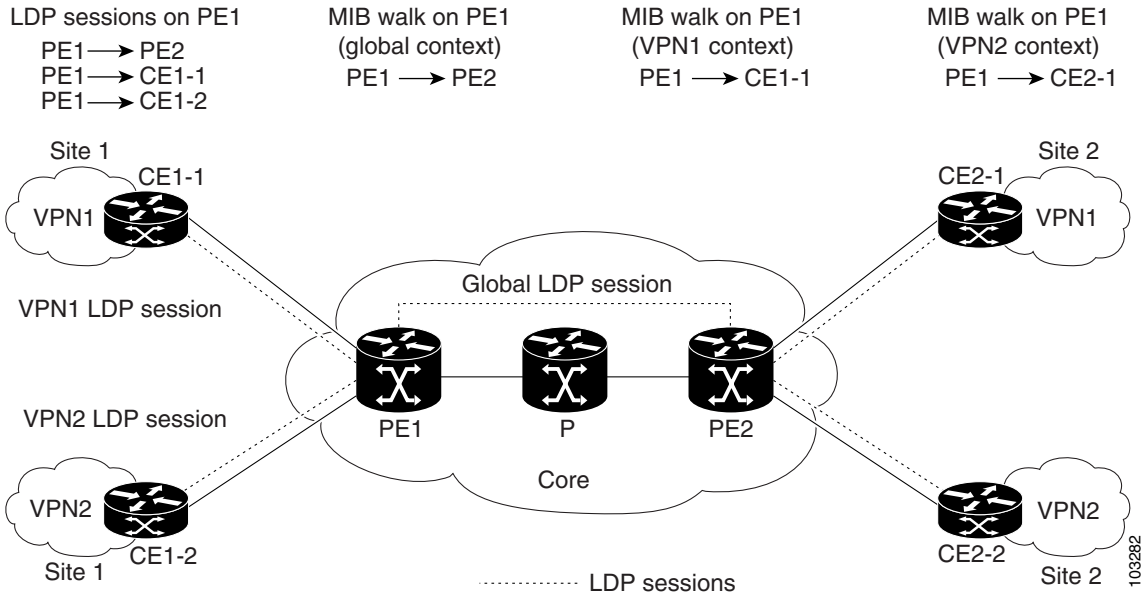
A MIB walk prior to this Cisco IOS release displayed only global session information.

With the VPN Aware LDP MIB enhancement in this Cisco IOS release, an SNMP query to the IETF MPLS-LDP-MIB supports both global and VPN contexts. This feature allows you to enter LDP queries on any VRF and on the core (global context). A query can differentiate between LDP sessions from different VPNs. LDP session information for a VPN stays in the context of that VPN. Therefore, the information from one VPN is not available to a user of a different VPN. The VPN Aware update to the LDP MIB also allows you to view LDP processes operating in a Carrier Supporting Carrier (CSC) network.

In an MPLS VPN, a service provider edge router (PE) might contain VRFs for several VPNs as well as a global routing table. To set up separate LDP processes for different VPNs on the same device, you need to configure each VPN with a unique securityName, contextName, and View-based Access Control Model (VACM) view. The VPN securityName must be configured for the IETF MPLS LDP MIB.

Figure 8 shows LDP sessions for a sample MPLS VPN network with the VPN Aware LDP MIB feature.

Figure 8 MPLS LDP Sessions with the VPN Aware LDP MIB Feature



With the VPN Aware LDP MIB feature, you can do MIB queries or MIB walks for an MPLS VPN LDP session or a global LDP session.

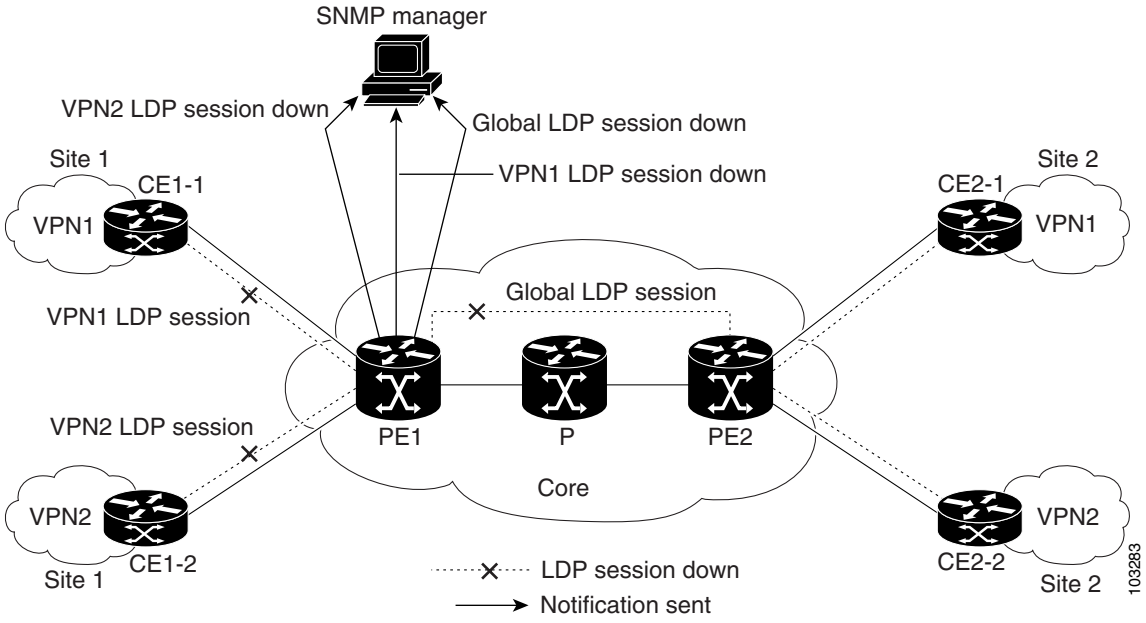
Note To verify LDP session information for a specific VPN, use the **show mpls ldp neighbor vrf** *vpn-name* **detail** command.

VPN Aware LDP MIB Notifications

Prior to Cisco IOS Release 12.0(11)ST, all notification messages for MPLS LDP sessions were sent to the same designated network management station (NMS) in the network. The notifications were enabled with the **snmp-server enable traps mpls ldp** command.

Figure 9 shows LDP notifications that were sent before the implementation of the VPN Aware LDP MIB feature.

Figure 9 LDP Notifications Sent Before the VPN Aware LDP MIB Feature



The VPN Aware LDP MIB feature supports LDP notifications for multiple LDP contexts for VPNs. LDP notifications can be generated for the core (global context) and for different VPNs. You can cause notifications be sent to different NMS hosts for different LDP contexts. LDP notifications associated with a specific VRF are sent to the NMS designated for that VRF. LDP global notifications are sent to the NMS configured to receive global traps.

To enable LDP context notifications for the VPN Aware LDP MIB feature, use either the SNMP object mplsLdpSessionsUpDownEnable (in the global LDP context only) or the following extended global configuration commands.

To enable LDP notifications for the global context, use the following commands:

```
PE-Router(config)# snmp-server host host-address traps community mpls-ldp

PE-Router(config)# snmp-server enable traps mpls ldp
```

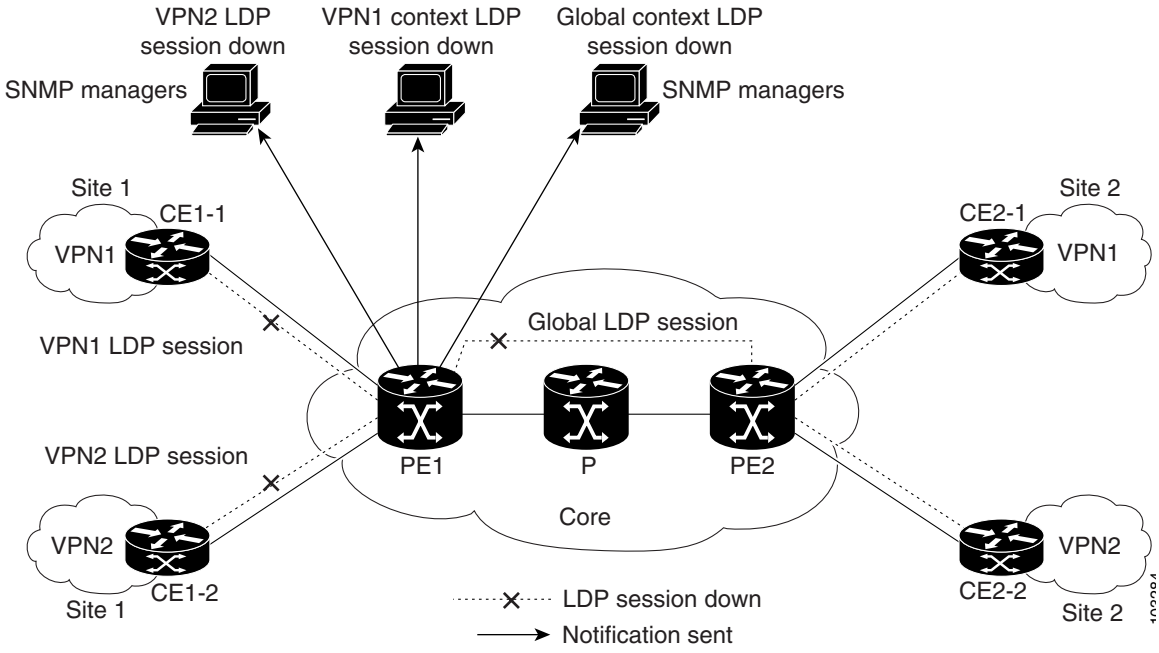
To enable LDP notifications for a VPN context, use the following commands:

```
PE-Router(config)# snmp-server host host-address vrf vrf-name version {v1|v2c|v3}
community community-string udp-port upd-port mpls-ldp

PE-Router(config)# snmp-server enable traps mpls ldp
```

Figure 10 shows LDP notifications with the VPN Aware LDP MIB feature.

Figure 10 LDP Notifications With the VPN Aware LDP MIB Feature



How to Configure MPLS LDP MIB Version 8 Upgrade

This section contains the following procedures:

- Enabling the SNMP Agent, page 25 (required)
- Enabling Cisco Express Forwarding, page 26 (required)
- Enabling MPLS Globally, page 27 (required)
- Enabling LDP Globally, page 28 (required)
- Enabling MPLS on an Interface, page 28 (required)
- Enabling LDP on an Interface, page 29 (required)
- Configuring a VPN Aware LDP MIB, page 30 (required)
- Verifying MPLS LDP MIB Version 8 Upgrade, page 35 (optional)

Enabling the SNMP Agent

Perform this task to enable the SNMP agent.

SUMMARY STEPS

1. **enable**
2. **show running-config**
3. **configure terminal**
4. **snmp-server community** *string* [**view** *view-name*] [**ro**] [*number*]

5. **end**
6. **write memory**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Router> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**show running-config** **Example:** `Router# show running-config`	Displays the running configuration of the router so that you can determine if an SNMP agent is already running on the device. If no SNMP information is displayed, continue with the next step. If any SNMP information is displayed, you can modify the information or change it as desired.
Step 3	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 4	**snmp-server community** *string* [**view** *view-name*] [**ro**] [*number*] **Example:** `Router(config)# snmp-server community public ro`	Configures read-only (ro) community strings for the MPLS LDP MIB. • The *string* argument functions like a password, permitting access to SNMP functionality on label switch routers (LSRs) in an MPLS network. • The optional **ro** keyword configures read-only (ro) access to the objects in the MPLS LDP MIB.
Step 5	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.
Step 6	**write memory** **Example:** `Router# write memory`	Writes the modified SNMP configuration into NVRAM of the router, permanently saving the SNMP settings.

Enabling Cisco Express Forwarding

Perform this task to enable Cisco Express Forwarding or distributed Cisco Express Forwarding.

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **ip cef distributed**
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.
Step 3	**ip cef distributed** **Example:** `Router(config)# ip cef distributed`	Enables distributed Cisco Express Forwarding.
Step 4	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.

Enabling MPLS Globally

Perform this task to enable MPLS globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls ip**
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	**mpls ip** **Example:** `Router(config)# mpls ip`	Enables MPLS forwarding of IPv4 packets along normally routed paths for the platform.
Step 4	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.

Enabling LDP Globally

Perform this task to enable LDP globally.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **mpls label protocol** {**ldp** | **tdp**}
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.
Step 3	**mpls label protocol** {**ldp** | **tdp**} **Example:** `Router(config)# mpls label protocol ldp`	Specifies the platform default label distribution protocol.
Step 4	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.

Enabling MPLS on an Interface

Perform this task to enable MPLS on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** [*type number*]
4. **mpls ip**
5. **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 number*] **Example:** `Router(config)# interface Ethernet 1`	Enters interface configuration mode. • The *type number* argument identifies the interface to be configured.
Step 4	**mpls ip** **Example:** `Router(config-if)# mpls ip`	Enables MPLS forwarding of IPv4 packets along normally routed paths for a particular interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.

Enabling LDP on an Interface

Perform this task to enable LDP on an interface.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** [*type number*]
4. **mpls label protocol** {**ldp** | **tdp** | **both**}
5. **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 number*] **Example:** `Router(config)# interface Ethernet 1`	Enters interface configuration mode. • The *type number* argument identifies the interface to be configured.
Step 4	**mpls label protocol** {**ldp** | **tdp** | **both**} **Example:** `Router(config-if)# mpls label protocol ldp`	Specifies the label distribution protocol to be used on a given interface.
Step 5	**end** **Example:** `Router(config-if)# end`	Exits to privileged EXEC mode.

Configuring a VPN Aware LDP MIB

To configure a VPN Aware LDP MIB, perform the following tasks:

- Configuring SNMP Support for a VPN, page 30
- Configuring an SNMP Context for a VPN, page 31
- Associating an SNMP VPN Context with SNMPv1 or SNMPv2, page 33

Configuring SNMP Support for a VPN

Perform this task to configure SNMP support for a Virtual Private Network (VPN) or a remote VPN.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] [**vrf** *vrf-name*]
4. **snmp-server engineID remote** *ip-address* [**udp-port** *udp-port-number*] [**vrf** *vrf-name*] *engineid-string*
5. **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	**snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [*notification-type*] [**vrf** *vrf-name*] **Example:** `Router(config)# snmp-server host example.com vrf trap-vrf`	Specifies the recipient of an SNMP notification operation and specifies the Virtual Private Network (VPN) routing and forwarding (VRF) instance table to be used for the sending of 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 80000009030000B064EFE100`	Configures a name for the remote SNMP engine on a router.
Step 5	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.

What to Do Next

Proceed to the "Configuring an SNMP Context for a VPN" section on page 31.

Configuring an SNMP Context for a VPN

Perform this task to configure an SNMP context for a VPN. This sets up a unique SNMP context for a VPN, which allows you to access the VPN's LDP session information.

SNMP Context

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 users of other VPNs on the same networking device.

VPN Route Distinguishers

A route distinguisher (RD) creates routing and forwarding tables for a VPN. Cisco IOS adds the RD to the beginning of the customer's IPv4 prefixes to change them into globally unique VPN-IPv4 prefixes.

Either the RD is an autonomous system number (ASN)-relative RD, in which case it is composed of an autonomous system number and an arbitrary number, or it is an IP-address-relative RD, in which case it is composed of an IP address and an arbitrary number. You can enter an RD in either of these formats:

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

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*
8. **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	**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 Virtual Private Network (VPN) routing and forwarding instance (VRF) table and enters VRF configuration mode.
Step 5	**rd** *route-distinguisher* **Example:** `Router(config-vrf)# rd 100:120`	Creates a VPN route distinguisher.

	Command or Action	Purpose
Step 6	**context** *context-name* **Example:** `Router(config-vrf)# context context1`	Associates an SNMP context with a particular VRF.
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.
Step 8	**end** **Example:** `Router(config)# end`	Exits to privileged EXEC mode.

What to Do Next

Proceed to the "Associating an SNMP VPN Context with SNMPv1 or SNMPv2" section on page 33.

Associating an SNMP VPN Context with SNMPv1 or SNMPv2

Perform this task to associate an SNMP VPN context with SNMPv1 or SNMPv2. This allows you to access LDP session information for a VPN using SNMPv1 or SNMPv2.

SNMPv1 or SNMPv2 Security

SNMPv1 and SNMPv2 are not as secure as SNMPv3. SNMP Versions 1 and 2 use plain text communities and do not perform the authentication or security checks that SNMP Version 3 performs.

To configure the VPN Aware LDP MIB 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 they come in from the configured VRF. If the community string contained in the incoming packet does not have an associated VRF, the packet is processed only if it came in through a non-VRF interface. This process prevents users outside the VPN from using a clear text community string to query the VPN data. However, this is not as secure as using SNMPv3.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **snmp-server user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*]
4. **snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*]
5. **snmp-server view** *view-name oid-tree* {**included** | **excluded**}
6. **snmp-server enable traps** [*notification-type*]

7. **snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [**notification-type**] [**vrf** *vrf-name*]
8. **snmp mib community-map** *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*] **target-list** *vpn-list-name*
9. **snmp mib target list** *vpn-list-name* {**vrf** *vrf-name* | **host** *ip-address*}
10. **no snmp-server trap authentication vrf**
11. **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 user** *username group-name* [**remote** *host* [**udp-port** *port*]] {**v1** | **v2c** | **v3** [**encrypted**] [**auth** {**md5** | **sha**} *auth-password*]} [**access** *access-list*] **Example:** `Router(config)# snmp-server user customer1 group1 v1`	Configures a new user to an SNMP group.
Step 4	**snmp-server group** *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*] [**read** *readview*] [**write** *writeview*] [**notify** *notifyview*] [**access** *access-list*] **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 and argument 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*] **Example:** `Router(config)# snmp-server enable traps`	Enables all SNMP notifications (traps or informs) available on your system.

	Command or Action	Purpose
Step 7	**snmp-server host** *host-address* [**traps** | **informs**] [**version** {**1** | **2c** | **3** [**auth** | **noauth** | **priv**]}] *community-string* [**udp-port** *port*] [**notification-type**] [**vrf** *vrf-name*] **Example:** `Router(config)# snmp-server host 10.0.0.1 vrf customer1 public udp-port 7002`	Specifies the recipient of an SNMP notification operation.
Step 8	**snmp mib community-map** *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*] **target-list** *vpn-list-name* **Example:** `Router(config)# snmp mib community-maps community1 context context1 target-list commAVpn`	Associates an SNMP community with an SNMP context, Engine ID, or security name.
Step 9	**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 10	**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.
Step 11	**exit** **Example:** `Router(config) exit`	Exits to privileged EXEC mode.

Verifying MPLS LDP MIB Version 8 Upgrade

Perform a MIB walk using your SNMP management tool to verify that the MPLS LDP MIB Version 8 Upgrade feature is functioning.

Configuration Examples for MPLS LDP MIB Version 8 Upgrade

This section provides the following configuration examples:

- MPLS LDP MIB Version 8 Upgrade Examples, page 36
- Configuring a VPN Aware SNMP Context for SNMPv1 or SNMPv2: Example, page 36

MPLS LDP MIB Version 8 Upgrade Examples

The following example shows how to enable an SNMP agent on the host NMS:

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

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

The following example shows how to enable SNMPv1 and SNMPv2C on the host NMS. The configuration permits any SNMP agent to access all MPLS LDP MIB objects that have read-only permission using the community string public.

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

The following example shows how to allow read-only access to all MPLS LDP MIB objects relating to members of access list 4 that specify the comaccess community string. No other SNMP agents will have access to any of the MPLS LDP MIB objects.

```
Router(config)# snmp-server community comaccess ro 4
```

The following example shows how to enable LDP globally and then on an interface:

```
Router# configure terminal

Enter configuration commands, one per line. End with CNTL/Z.

Router(config)# mpls label protocol ldp

Router(config)# interface Ethernet1

Router(config-if)# mpls label protocol ldp

Router(config-if)# end
```

Configuring a VPN Aware SNMP Context for SNMPv1 or SNMPv2: Example

The following configuration example shows how to configure a VPN Aware SNMP context for the MPLS LDP MIB Version 8 with SNMPv1 or SNMPv2:

```
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 10.0.0.0 255.255.0.0

interface Ethernet3/2
 description Belongs to VPN B
 ip vrf forwarding CustomerB
 ip address 10.0.0.1 255.255.0.0

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 10.0.0.3 vrf CustomerA commA udp-port 7002
snmp-server host 10.0.0.4 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

The following sections provide references related to the MPLS LDP MIB Version 8 Upgrade feature.

Related Documents

Related Topic	Document Title
MPLS LDP configuration tasks	*MPLS Label Distribution Protocol (LDP)*
A description of SNMP agent support in Cisco IOS software for the MPLS Label Switching Router MIB (MPLS-LSR-MIB)	*MPLS Label Switching Router MIB*
A description of SNMP agent support in Cisco IOS software for the MPLS Traffic Engineering MIB (MPLS TE MIB)	*MPLS Traffic Engineering (TE) MIB*
Configuration tasks for MPLS ATM network enhancements	*MPLS Scalability Enhancements for the ATM LSR*

Related Topic	Document Title
MPLS automatic bandwidth adjustment configuration tasks	*MPLS Traffic Engineering (TE)—Automatic Bandwidth Adjustment for TE Tunnels*
A description of MPLS differentiated types of service across an MPLS network	*MPLS Class of Service*
SNMP commands	*Cisco IOS Network Management Command Reference,* Release 12.4T
SNMP configuration	"Configuring SNMP Support" chapter in the *Cisco IOS Network Management Configuration Guide,* Release 12.4
SNMP support for VPNs	*SNMP Notification Support for VPNs*
SNMP context support for VPNs configuration tasks	*SNMP Support over VPNs—Context Based Access Control*

Standards

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

MIBs

MIBs	MIBs Link
• MPLS Label Distribution Protocol MIB (draft-ietf-mpls-ldp-mib-08.txt) • SNMP-VACM-MIB The View-based Access Control Model (ACM) MIB for SNMP	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

RFCs	Title
RFC 2233 The LDP implementation supporting the MPLS LDP MIB fully complies with the provisions of Section 10 of RFC 2026, which, in effect, states that the implementation of LDP is recommended for network devices that perform MPLS forwarding along normally routed paths, as determined by destination-based routing protocols.	*Interfaces MIB*

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/techsupport

Command Reference

This section documents only commands that are new or modified.

- **context**
- **show mpls ldp neighbor**
- **snmp mib community-map**
- **snmp mib target list**
- **snmp-server community**
- **snmp-server context**
- **snmp-server enable traps (MPLS)**
- **snmp-server group**
- **snmp-server host**
- **snmp-server trap authentication vrf**

context

To associate a Simple Network Management Protocol (SNMP) context with a particular virtual private network (VPN) routing and forwarding instance (VRF), use the **context** command in VRF configuration mode. To disassociate an SNMP context from a VPN, use the **no** form of this command.

context *context-name*

no context *context-name*

Syntax Description

context-name	Name of the SNMP VPN context, up to 32 characters.

Command Default

No SNMP contexts are associated with VPNs.

Command Modes

VRF configuration (config-vrf)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SRB	Support for IPv6 was added.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Before you use this command to associate an SNMP context with a VPN, you must do the following:

- Issue the **snmp-server context** command to create an SNMP context
- Associate a VPN with a context so that the specific MIB data for that VPN exists in that context.
- Associate a VPN group with the context of the VPN using the **snmp-server group** command with the **context** *context-name* keyword and argument.

SNMP contexts provide VPN users with a secure way of accessing MIB data. When a VPN is associated with a context, MIB data for that VPN exists in that context. Associating a VPN with a context helps enable 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 users of other VPNs on the same networking device.

A route distinguisher (RD) is required when you configure an SNMP context. An RD creates routing and forwarding tables and specifies the default route distinguisher for a VPN. The RD is added to the beginning of a IPv4 prefix to make it globally unique. An RD is either ASN relative, which means it is composed of an autonomous system number and an arbitrary number, or it is IP address relative and composed of an IP address and an arbitrary number.

Examples

The following example shows how to create an SNMP context named context1 and associate the context with the VRF named vrf1:

```
Router(config)# snmp-server context1
Router(config)# ip vrf vrf1
Router(config-vrf)# rd 100:120
Router(config-vrf)# context context1
```

Related Commands

Command	Description
ip vrf	Enters VRF configuration mode for the configuration of a VRF.
snmp mib community-map	Associates an SNMP community with an SNMP context, engine ID, or security name.
snmp mib target list	Creates a list of target VRFs and hosts to associate with an SNMP v1 or v2c community.
snmp-server context	Creates an SNMP context.
snmp-server group	Configures a new SNMP group, or a table that maps SNMP users to SNMP views.
snmp-server trap authentication vrf	Controls VRF-specific SNMP authentication failure notifications.
snmp-server user	Configures a new user to an SNMP group.

show mpls ldp neighbor

To display the status of Label Distribution Protocol (LDP) sessions, use the **show mpls ldp neighbor** command in user EXEC or privileged EXEC mode.

show mpls ldp neighbor [**vrf** *vrf-name* | **all**] [*address* | *interface*] [**detail**] [**graceful-restart**]

Syntax Description

vrf *vrf-name*	(Optional) Displays the LDP neighbors for the specified Virtual Private Network (VPN) routing and forwarding (VRF) instance (*vrf-name*).
all	(Optional) Displays LDP neighbor information for all VPNs, including those in the default routing domain.
address	(Optional) Identifies the neighbor with this IP address.
interface	(Optional) Defines the LDP neighbors accessible over this interface.
detail	(Optional) Displays information in long form.
graceful-restart	(Optional) Displays per-neighbor graceful restart information.

Command Default

This command displays information about LDP neighbors for the default routing domain if you do not specify the optional **vrf** keyword.

Command Modes

User EXEC (>)
Privileged EXEC (#)

Command History

Release	Modification
11.1CT	This command was introduced.
12.0(10)ST	The command was modified to reflect Multiprotocol Label Switching (MPLS) Internet Engineering Task Force (IETF) command syntax and terminology.
12.0(14)ST	This command was modified to reflect MPLS VPN support for LDP and the **vrf** and **all** keywords were added.
12.1(8a)E	This command was integrated into Cisco IOS Release 12.1(8a)E.
12.2(2)T	This command was integrated into Cisco IOS Release 12.2(2)T.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.0(26)S	The **detail** keyword was updated to display information about inbound filtering.
12.2(25)S	The **graceful-restart** keyword was added.
12.3(14)T	The command output was updated so that the **detail** keyword displays information about MPLS LDP Session Protection.
12.2(18)SXE	This command was integrated into Cisco IOS Release 12.2(18)SXE.
12.2(28)SB	The **detail** keyword was updated to include Message Digest 5 (MD5) password information and the command was implemented on the Cisco 10000 Series Routers.

Release	Modification
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **show mpls ldp neighbor** command can provide information about all LDP neighbors, or the information can be limited to the following:

- Neighbor with specific IP address
- LDP neighbors known to be accessible over a specific interface

Note This command displays information about LDP and Tag Distribution Protocol (TDP) neighbor sessions.

Examples

For explanations of the significant fields shown in the displays, see Table 11.

The following is sample output from the **show mpls ldp neighbor** command:

```
Router# show mpls ldp neighbor

Peer LDP Ident: 10.0.7.7:2; Local LDP Ident 10.1.1.1:1
        TCP connection: 10.0.7.7:11032 - 10.1.1.1:646
        State: Oper; Msgs sent/rcvd: 5855/6371; Downstream on demand
        Up time: 13:15:09
        LDP discovery sources:
          ATM3/0.1
Peer LDP Ident: 10.1.1.1:0; Local LDP Ident 10.1.1.1:0
        TCP connection: 10.1.1.1:646 - 10.1.1.1:11006
        State: Oper; Msgs sent/rcvd: 4/411; Downstream
        Up time: 00:00:52
        LDP discovery sources:
          Ethernet1/0/0
        Addresses bound to peer LDP Ident:
          10.0.0.29       10.1.1.1        10.0.0.199      10.10.1.1
          10.205.0.9
```

The following is sample output from the **show mpls ldp neighbor** command, in which duplicate addresses are detected. They indicate an error because a given address should be bound to only one peer.

```
Router# show mpls ldp neighbor

Peer LDP Ident: 10.0.7.7:2; Local LDP Ident 10.1.1.1:1
        TCP connection: 10.0.7.7:11032 - 10.1.1.1:646
        State: Oper; Msgs sent/rcvd: 5855/6371; Downstream on demand
        Up time: 13:15:09
        LDP discovery sources:
          ATM3/0.1
Peer LDP Ident: 10.1.1.1:0; Local LDP Ident 10.1.1.1:0
        TCP connection: 10.1.1.1:646 - 10.1.1.1:11006
        State: Oper; Msgs sent/rcvd: 4/411; Downstream
        Up time: 00:00:52
        LDP discovery sources:
          Ethernet1/0/0
        Addresses bound to peer LDP Ident:
          10.0.0.29, 10.1.1.1, 10.0.0.199, 10.10.1.1
          10.205.0.9
        Duplicate Addresses advertised by peer:
          10.10.8.111
```

The following is sample output from the **show mpls ldp neighbor vrf vpn10** command, which displays the LDP neighbor information for the specified VPN routing and forwarding instance named vpn10:

```
Router# show mpls ldp neighbor vrf vpn10

Peer LDP Ident:10.14.14.14:0; Local LDP Ident 10.29.0.2:0
        TCP connection:10.14.14.14:646 - 10.29.0.2:11384
        State:Oper; Msgs sent/rcvd:1423/800; Downstream
        Up time:02:38:11
        LDP discovery sources:
          ATM3/0/0.10
        Addresses bound to peer LDP Ident:
          10.3.36.9       10.7.0.1        10.14.14.14     10.13.0.1
          10.15.0.1       10.17.0.1       10.19.0.1       10.21.0.1
          10.23.0.1       10.25.0.1       10.27.0.1       10.29.0.1
          10.31.0.1       10.33.0.1       10.35.0.1       10.37.0.1
          10.39.0.1       10.41.0.1       10.43.0.1       10.45.0.1
          10.47.0.1       10.49.0.1       10.51.0.1       10.53.0.1
          10.55.0.1       10.57.0.1       10.59.0.1       10.61.0.1
          10.63.0.1       10.65.0.1       10.67.0.1       10.69.0.1
          10.71.0.1       10.73.0.1       10.75.0.1       10.77.0.1
          10.79.0.1       10.81.0.1       10.83.0.1       10.85.0.1
          10.87.0.1       10.89.0.1       10.91.0.1       10.93.0.1
          10.95.0.1       10.97.0.1       10.99.0.1       10.101.0.1
          10.103.0.1      10.105.0.1      10.107.0.1      10.109.0.1
          10.4.0.2        10.3.0.2
```

The following shows sample output from the **show mpls ldp neighbor detail** command, which displays information about inbound filtering:

```
Router# show mpls ldp neighbor vrf vpn1 detail

Peer LDP Ident: 10.13.13.13:0; Local LDP Ident 10.33.0.2:0
   TCP connection: 10.13.13.13:646 - 10.33.0.2:31581
   State: Oper; Msgs sent/rcvd: 11/10; Downstream; Last TIB rev sent 13
   Up time: 00:02:25; UID: 26; Peer Id 0;
   LDP discovery sources:
    Ethernet1/0/2; Src IP addr: 10.33.0.1
     holdtime: 15000 ms, hello interval: 5000 ms
   Addresses bound to peer LDP Ident:
    10.3.105.1       10.13.13.13     10.33.0.1
   Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
   LDP inbound filtering accept acl:1
 Peer LDP Ident: 10.14.14.14:0; Local LDP Ident 10.33.0.2:0
   TCP connection: 10.14.14.14:646 - 10.33.0.2:31601
   State: Oper; Msgs sent/rcvd: 10/9; Downstream; Last TIB rev sent 13
   Up time: 00:01:17; UID: 29; Peer Id 3;
   LDP discovery sources:
Ethernet1/0/3; Src IP addr: 10.33.0.1
     holdtime: 15000 ms, hello interval: 5000 ms
   Addresses bound to peer LDP Ident:
    10.3.104.1       10.14.14.14     10.32.0.1
   Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
   LDP inbound filtering accept acl:1
```

The following is sample output from the **show mpls ldp neighbor all** command, which displays the LDP neighbor information for all VPN routing and forwarding instances, including those in the default routing domain. In this example, note that the same neighbor LDP ID (10.14.14.14) appears in all the listed VRF interfaces, highlighting the fact that the same IP address can coexist in different VPN routing and forwarding instances.

```
Router# show mpls ldp neighbor all
```

```
Peer TDP Ident:10.11.11.11:0; Local TDP Ident 10.12.12.12:0
        TCP connection:10.11.11.11:711 - 10.12.12.12:11003
        State:Oper; PIEs sent/rcvd:185/187; Downstream
        Up time:02:40:02
        TDP discovery sources:
          ATM1/1/0.1
        Addresses bound to peer TDP Ident:
          10.3.38.3       10.1.0.2        10.11.11.11
VRF vpn1:
    Peer LDP Ident:10.14.14.14:0; Local LDP Ident 10.7.0.2:0
        TCP connection:10.14.14.14:646 - 10.7.0.2:11359
        State:Oper; Msgs sent/rcvd:952/801; Downstream
        Up time:02:38:49
        LDP discovery sources:
          ATM3/0/0.1
        Addresses bound to peer LDP Ident:
          10.3.36.9       10.7.0.1        10.14.14.14     10.13.0.1
          10.15.0.1       10.17.0.1       10.19.0.1       10.21.0.1
          10.23.0.1       10.25.0.1       10.27.0.1       10.29.0.1
          10.31.0.1       10.33.0.1       10.35.0.1       10.37.0.1
          10.39.0.1       10.41.0.1       10.43.0.1       10.45.0.1
          10.47.0.1       10.49.0.1       10.51.0.1       10.53.0.1
          10.55.0.1       10.57.0.1       10.59.0.1       10.61.0.1
          10.63.0.1       10.65.0.1       10.67.0.1       10.69.0.1
          10.71.0.1       10.73.0.1       10.75.0.1       10.77.0.1
          10.79.0.1       10.81.0.1       10.83.0.1       10.85.0.1
          10.87.0.1       10.89.0.1       10.91.0.1       10.93.0.1
          10.95.0.1       10.97.0.1       10.99.0.1       10.101.0.1
          10.103.0.1      10.105.0.1      10.107.0.1      10.109.0.1
          10.4.0.2        10.3.0.2
VRF vpn2:
    Peer LDP Ident:10.14.14.14:0; Local LDP Ident 10.13.0.2:0
        TCP connection:10.14.14.14:646 - 10.13.0.2:11361
        State:Oper; Msgs sent/rcvd:964/803; Downstream
        Up time:02:38:50
        LDP discovery sources:
          ATM3/0/0.2
        Addresses bound to peer LDP Ident:
          10.3.36.9       10.7.0.1        10.14.14.14     10.13.0.1
          10.15.0.1       10.17.0.1       10.19.0.1       10.21.0.1
          10.23.0.1       10.25.0.1       10.27.0.1       10.29.0.1
          10.31.0.1       10.33.0.1       10.35.0.1       10.37.0.1
          10.39.0.1       10.41.0.1       10.43.0.1       10.45.0.1
          10.47.0.1       10.49.0.1       10.51.0.1       10.53.0.1
          10.55.0.1       10.57.0.1       10.59.0.1       10.61.0.1
          10.63.0.1       10.65.0.1       10.67.0.1       10.69.0.1
          10.71.0.1       10.73.0.1       10.75.0.1       10.77.0.1
          10.79.0.1       10.81.0.1       10.83.0.1       10.85.0.1
          10.87.0.1       10.89.0.1       10.91.0.1       10.93.0.1
          10.95.0.1       10.97.0.1       10.99.0.1       10.101.0.1
          10.103.0.1      10.105.0.1      10.107.0.1      10.109.0.1
          10.4.0.2        10.3.0.2
VRF vpn3:
    Peer LDP Ident:10.14.14.14:0; Local LDP Ident 10.15.0.2:0
        TCP connection:10.14.14.14:646 - 10.15.0.2:11364
        State:Oper; Msgs sent/rcvd:1069/800; Downstream
        Up time:02:38:52
        LDP discovery sources:
          ATM3/0/0.3
        Addresses bound to peer LDP Ident:
          10.3.36.9       10.17.0.1       10.14.14.14     10.13.0.1
          10.15.0.1       10.17.0.1       10.19.0.1       10.21.0.1
          10.23.0.1       10.25.0.1       10.27.0.1       10.29.0.1
          10.31.0.1       10.33.0.1       10.35.0.1       10.37.0.1
```

```
          10.39.0.1       10.41.0.1       10.43.0.1       10.45.0.1
          10.47.0.1       10.49.0.1       10.51.0.1       10.53.0.1
          10.55.0.1       10.57.0.1       10.59.0.1       10.61.0.1
          10.63.0.1       10.65.0.1       10.67.0.1       10.69.0.1
          10.71.0.1       10.73.0.1       10.75.0.1       10.77.0.1
          10.79.0.1       10.81.0.1       10.83.0.1       10.85.0.1
          10.87.0.1       10.89.0.1       10.91.0.1       10.93.0.1
          10.95.0.1       10.97.0.1       10.99.0.1       10.101.0.1
          10.103.0.1      10.105.0.1      10.107.0.1      10.109.0.1
          10.4.0.2        10.3.0.2
VRF vpn4:
    Peer LDP Ident:10.14.14.14:0; Local LDP Ident 10.17.0.2:0
        TCP connection:10.14.14.14:646 - 10.17.0.2:11366
        State:Oper; Msgs sent/rcvd:1199/802; Downstream
```

The following example shows the Graceful Restart status of the LDP neighbors:

```
Router# show mpls ldp neighbor graceful-restart

Peer LDP Ident: 10.20.20.20:0; Local LDP Ident 10.17.17.17:0
   TCP connection: 10.20.20.20:16510 - 10.17.17.17:646
    State: Oper; Msgs sent/rcvd: 8/18; Downstream
    Up time: 00:04:39
    Graceful Restart enabled; Peer reconnect time (msecs): 120000
Peer LDP Ident: 10.19.19.19:0; Local LDP Ident 10.17.17.17:0
    TCP connection: 10.19.19.19:11007 - 10.17.17.17:646
    State: Oper; Msgs sent/rcvd: 8/38; Downstream
    Up time: 00:04:30
    Graceful Restart enabled; Peer reconnect time (msecs): 120000
```

The following sample output from the **show mpls ldp neighbor detail** command displays information about the MD5 password configuration:

```
Router# show mpls ldp neighbor detail

    Peer LDP Ident: 10.3.3.3:0; Local LDP Ident 10.1.1.1:0
        TCP connection: 10.3.3.3:11018 - 10.1.1.1:646
        Password: required, neighbor, in use
        State: Oper; Msgs sent/rcvd: 167/167; Downstream; Last TIB rev sent 9
        Up time: 02:24:02; UID: 5; Peer Id 3;
        LDP discovery sources:
          Targeted Hello 10.1.1.1 -> 10.3.3.3, passive;
            holdtime: 90000 ms, hello interval: 10000 ms
        Addresses bound to peer LDP Ident:
          10.3.3.3        10.0.30.3
        Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
    Peer LDP Ident: 10.4.4.4:0; Local LDP Ident 10.1.1.1:0
        TCP connection: 10.4.4.4:11017 - 10.1.1.1:646
        Password: not required, none, stale
        State: Oper; Msgs sent/rcvd: 9/9; Downstream; Last TIB rev sent 9
        Up time: 00:05:35; UID: 6; Peer Id 1;
        LDP discovery sources:
          Ethernet1/0; Src IP addr: 10.0.20.4
            holdtime: 15000 ms, hello interval: 5000 ms
        Addresses bound to peer LDP Ident:
          10.0.40.4       10.4.4.4        10.0.20.4
        Peer holdtime: 180000 ms; KA interval: 60000 ms; Peer state: estab
```

Table 11 describes the significant fields shown in the displays.

Table 11 show mpls ldp neighbor Field Descriptions

Field	Description
Peer LDP Ident	LDP (or TDP) identifier of the neighbor (peer) for this session.
Local LDP Ident	LDP (or TDP) identifier for the local label switch router (LSR) for this session.
TCP connection	TCP connection used to support the LDP session, shown in the following format: • peer IP address.peer port • local IP address.local port
Password	Indicates if password protection is being used. Password status is as follows: • Required or not required—Indicates whether password configuration is required. • Neighbor, none, option #, or fallback—Indicates the password source when the password was configured. • In use (current) or stale (previous)—Indicates the current LDP session password usage status.
State	State of the LDP session. Generally, this is Oper (operational), but transient is another possible state.
Msgs sent/rcvd	Number of LDP messages sent to and received from the session peer. The count includes the transmission and receipt of periodic keepalive messages, which are required for maintenance of the LDP session.
Downstream on demand	Indicates that the Downstream on Demand method of label distribution is being used for this LDP session. When the Downstream on Demand method is used, an LSR advertises its locally assigned (incoming) labels to its LDP peer only when the peer requests them.
Downstream	Indicates that the downstream method of label distribution is being used for this LDP session. When the downstream method is used, an LSR advertises all of its locally assigned (incoming) labels to its LDP peer (subject to any configured access list restrictions).
Up time	Length of time (in hours, minutes, seconds) the LDP session has existed.
Graceful Restart enabled	Indicates whether the LDP session has Graceful Restart enabled.
Peer reconnect time	The length of time, in milliseconds (msecs), the peer router waits for a router to reconnect.
LDP discovery sources	Sources of LDP discovery activity that led to the establishment of this LDP session.
Targeted Hello	Lists the platforms to which targeted hello messages are being sent: • The active field indicates that this LSR has initiated targeted hello messages. • The passive field indicates that the neighbor LSR has initiated targeted hello messages and that this LSR is configured to respond to the targeted hello messages from the neighbor.

Table 11 show mpls ldp neighbor Field Descriptions (continued)

Field	Description
holdtime	Period of time, in milliseconds (ms), a discovered LDP neighbor is remembered without receipt of an LDP hello message from the neighbor.
hello interval	Period of time, in milliseconds (ms), between the sending of consecutive hello messages.
Addresses bound to peer LDP Ident	Known interface addresses of the LDP session peer. These are addresses that might appear as "next hop" addresses in the local routing table. They are used to maintain the Label Forwarding Information Base (LFIB).
Duplicate Addresses advertised by peer	IP addresses that are bound to another peer. They indicate an error because a given address should be bound to only one peer.
Peer holdtime	The time, in milliseconds (ms), that the neighbor session is retained without the receipt of an LDP message from the neighbor.
KA Interval	Keep Alive Interval. The amount of time, in milliseconds (ms), that a router lets pass without sending an LDP message to its neighbor. If this time elapses and the router has nothing to send, it sends a Keep Alive message.
Peer state	State of the peer; estab means established.
LDP inbound filtering accept acl:1	Access list that is permitted for inbound label binding filtering.

Related Commands

Command	Description
show mpls ldp discovery	Displays the status of the LDP discovery process.

snmp mib community-map

To associate a Simple Network Management Protocol (SNMP) community with an SNMP context, engine ID, or security name, use the **snmp mib community-map** command in global configuration mode. To change an SNMP community mapping to its default mapping, use the **no** form of this command.

snmp mib community-map *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*] [**target-list** *vpn-list-name*]

no snmp mib community-map *community-name* [**context** *context-name*] [**engineid** *engine-id*] [**security-name** *security-name*] [**target-list** *vpn-list-name*]

Syntax Description

community-name	String that identifies the SNMP community.
context	(Optional) Specifies that an SNMP context name is mapped to the SNMP community.
context-name	(Optional) String that identifies the name of the SNMP context.
engineid	(Optional) Specifies that an SNMP engine ID is mapped to the SNMP community.
engine-id	(Optional) String that identifies the SNMP engine ID. Default is the local engine ID
security-name	(Optional) Specifies that a security name is mapped to the SNMP community.
security-name	(Optional) String that identifies the SNMP security name. Default is the community name
target-list	(Optional) Specifies that a VPN routing and forwarding (VRF) list is mapped to the SNMP community.
vpn-list-name	(Optional) String value that should correspond to the list name used in the **snmp mib target list** command.

Command Default

No SNMP communities and contexts are associated.

Command Modes

Global configuration (config)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use this command to create a mapping between an SNMP community and an SNMP context, engine ID, or security name that is different from the default settings.

Use the **snmp-server community** command to configure an SNMP community. When an SNMP community is associated with an SNMP context and a request is made from this community, the request is applied to the context. You also can use the **snmp mib community-map** command to specify the source address validation for an SNMP community by associating a list of target VRFs. The target VRF list specifies the valid host or hosts for this SNMP community.

Examples

The following example shows how to create an SNMP community named community1 and associate it with an SNMP context named context1:

```
Router(config)# snmp-server community community1
Router(config)# snmp mib community-map community1 context context1
```

The following example shows a mapping of community A (commA) to VPN list commAvpn and community B (commB) to VPN list commBvpn:

```
Router(config)# snmp mib community-map commA context A target-list commAvpn
Router(config)# snmp mib community-map commB context B target-list commBvpn
Router(config)# snmp mib target list commAvpn vrf CustomerA
Router(config)# snmp mib target list commBvpn vrf CustomerB
```

Related Commands

Command	Description
context	Associates an SNMP context with a particular VPN.
snmp-server community	Sets up the community access string to permit access to the SNMP.

snmp mib target list

To create a list of target virtual private network (VPN) routing and forwarding (VRF) instance and hosts to associate with a Simple Network Management Protocol (SNMP) community, use the **snmp mib target list** command in global configuration mode. To delete the list of VRF instances and hosts or to delete a particular VRF or host from the list, use the **no** form of this command.

snmp mib target list *vpn-list-name* {**vrf** *vrf-name* | **host** *ip-address*}

no snmp mib target list *vpn-list-name* {**vrf** *vrf-name* | **host** *ip-address*}

Syntax Description

vpn-list-name	Name of the target list.
vrf	Adds a specified VRF to the target list.
vrf-name	Name of a VRF to include in the list.
host	Adds a specified host to the target list.
ip-address	IP address of the host.

Command Default

No target list is created.

Command Modes

Global configuration (config)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31) SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

Use this command when using SNMPv1 or SNMPv2 in a VPN environment to configure a list of VRFs or hosts for source address validation. Configuring the target list ensures that the community is valid only if the incoming packet is received from a VRF or host on the target list.

- Only the following MIBs are context aware and all the tables in these MIBs can be polled:
  - CISCO-IPSEC-FLOW-MONITOR-MIB (Cisco IOS Release 12.4T and later)
  - CISCO-IPSEC-MIB (Cisco IOS Release 12.4T and later)
  - CISCO-PING-MIB
  - IP-FORWARD-MIB
  - MPLS-LDP-MIB

- Currently, 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).

Note It is recommended that you use SNMPv3 with the authNoPriv or higher level of security when using SNMP in a VPN environment.

Examples

The following example shows how to add a target list named target1 and add a VRF named vrf1 to the newly created target list:

```
Router(config)# snmp mib target list target1 vrf vrf1
```

Related Commands

Command	Description
snmp mib community-map	Associates an SNMP community with an SNMP context, engine ID, or security name.

snmp-server community

To set up the community access string to permit access to the Simple Network Management Protocol (SNMP), use the **snmp-server community** command in global configuration mode. To remove the specified community string, use the **no** form of this command.

snmp-server community *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*]

no snmp-server community *string*

Syntax Description

string	Community string that consists of 1 to 32 alphanumeric characters and functions much like a password, permitting access to SNMP. Blank spaces are not permitted in the community string. **Note** The @ symbol is used for delimiting the context information. Avoid using the @ symbol as part of the SNMP community string when configuring this command.
view	(Optional) Specifies a previously defined view. The view defines the objects available to the SNMP community.
view-name	(Optional) Name of a previously defined view.
ro	(Optional) Specifies read-only access. Authorized management stations can retrieve only MIB objects.
rw	(Optional) Specifies read-write access. Authorized management stations can both retrieve and modify MIB objects.
ipv6	(Optional) Specifies an IPv6 named access list.
nacl	(Optional) IPv6 named access list.
access-list-number	(Optional) Integer from 1 to 99 that specifies a standard access list of IP addresses or a string (not to exceed 64 characters) that is the name of a standard access list of IP addresses allowed access to the SNMP agent. Alternatively, an integer from 1300 to 1999 that specifies a list of IP addresses in the expanded range of standard access list numbers that are allowed to use the community string to gain access to the SNMP agent.

Command Default

An SNMP community string permits read-only access to all objects.

Note If the **snmp-server community** command is not used during the SNMP configuration session, the command will automatically be added to the configuration after the **snmp host** command is used. In this case, the default password (*string*) for the **snmp-server community** command will be taken from the **snmp host** command.

Command Modes

Global configuration (config)

Command History

Release	Modification
10.0	This command was introduced.
12.0(14)ST	This command was integrated into Cisco IOS Release 12.0(14)ST.
12.0(17)S	This command was integrated into Cisco IOS Release 12.0(17)S.
12.2(14)S	This command was integrated into Cisco IOS Release 12.2(14)S.
12.3(2)T	The access list values were enhanced to support the expanded range of standard access list values and to support named standard access lists.
12.0(27)S	The **ipv6** *nacl* keyword and argument pair was added to support assignment of IPv6 named access lists. This keyword and argument pair is not supported in Cisco IOS 12.2S releases.
12.3(14)T	The **ipv6** *nacl* keyword and argument pair was integrated into Cisco IOS Release 12.3(14)T to support assignment of IPv6 named access lists. This keyword and argument pair is not supported in Cisco IOS 12.2S releases.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **no snmp-server** command disables all versions of SNMP (SNMPv1, SNMPv2C, SNMPv3).

The first **snmp-server** command that you enter enables all versions of SNMP.

To configure SNMP community strings for the MPLS LDP MIB, use the **snmp-server community** command on the host network management station (NMS).

The **snmp-server community** command can be used to specify only an IPv6 named access list, only an IPv4 access list, or both. For you to configure both IPv4 and IPv6 access lists, the IPv6 access list must appear first in the command statement.

Note The @ symbol is used as a delimiter between the community string and the context in which it is used. For example, specific VLAN information in BRIDGE-MIB may be polled using community@VLAN_ID (for example, public@100) where 100 is the VLAN number. Avoid using the @ symbol as part of the SNMP community string when configuring this command.

Examples

The following example shows how to set the read/write community string to newstring:

```
Router(config)# snmp-server community newstring rw
```

The following example shows how to allow read-only access for all objects to members of the standard named access list lmnop that specify the comaccess community string. No other SNMP managers have access to any objects.

```
Router(config)# snmp-server community comaccess ro lmnop
```

The following example shows how to assign the string comaccess to SNMP, allow read-only access, and specify that IP access list 4 can use the community string:

```
Router(config)# snmp-server community comaccess ro 4
```

The following example shows how to assign the string manager to SNMP and allow read-write access to the objects in the restricted view:

```
Router(config)# snmp-server community manager view restricted rw
```

The following example shows how to remove the community comaccess:

```
Router(config)# no snmp-server community comaccess
```

The following example shows how to disable all versions of SNMP:

```
Router(config)# no snmp-server
```

The following example shows how to configure an IPv6 access list named list1 and links an SNMP community string with this access list:

```
Router(config)# ipv6 access-list list1
Router(config-ipv6-acl)# permit ipv6 any any
Router(config-ipv6-acl)# exit
Router(config)# snmp-server community comaccess rw ipv6 list1
```

Related Commands

Command	Description
access-list	Configures the access list mechanism for filtering frames by protocol type or vendor code.
snmp-server enable traps	Enables the router to send SNMP notification messages to a designated network management workstation.
snmp-server host	Specifies the targeted recipient of an SNMP notification operation.
snmp-server view	Creates or updates a view entry.

snmp-server context

To create a Simple Network Management Protocol (SNMP) context, use the **snmp-server context** command in global configuration mode. To delete an SNMP context, use the **no** form of this command.

snmp-server context *context-name*

no snmp-server context *context-name*

Syntax Description

context-name	Name of the SNMP context being created.

Command Default

No SNMP contexts are configured.

Command Modes

Global configuration (config)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

When you use the **no snmp-server context** command, all SNMP instances in that context are deleted.

A route distinguisher (RD) is required when you configure an SNMP context. An RD creates routing and forwarding tables and specifies the default route distinguisher for a VPN. The RD is added to the beginning of a IPv4 prefix to make it globally unique. An RD is either ASN relative, which means it is composed of an autonomous system number and an arbitrary number, or it is IP address relative and composed of an IP address and an arbitrary number.

Examples

The following example shows how to create an SNMP context named contextA and associate it with a virtual private network (VPN) routing and forwarding (VRF) instance named CustomerA:

```
Router(config)# snmp-server context contextA
Router(config)# ip vrf CustomerA
Router(config-vrf)# rd 100:120
Router(config-vrf)# context contextA
```

Related Commands

Command	Description
context	Associates an SNMP context with a particular VRF.

snmp-server enable traps (MPLS)

To enable a label switch router (LSR) to send Simple Network Management Protocol (SNMP) notifications or informs to an SNMP host, use the **snmp-server enable traps** command in global configuration mode. To disable notifications or informs, use the **no** form of this command.

snmp-server enable traps [*notification-type*] [*notification-option*]

no snmp-server enable traps [*notification-type*] [*notification-option*]

Syntax Description

notification-type — (Optional) Specifies the particular type of SNMP notification(s) to be enabled on the LSR. If a notification type is not specified, all SNMP notifications applicable to the LSR are enabled and sent to the SNMP host. Any one or all of the following keywords can be specified in any combination as the *notification-type* (family name) in the **snmp-server enable traps** command:

- **bgp**—Sends Border Gateway Protocol (BGP) state change notifications.
- **config**—Sends configuration notifications.
- **entity**—Sends entity MIB modification notifications.
- **envmon**—Sends Cisco enterprise-specific environmental monitor notifications whenever certain environmental thresholds are exceeded. *Notification-option* arguments (below) can be specified in combination with this keyword.
- **frame-relay**—Sends Frame Relay notifications.
- **hsrp**—Sends Hot Standby Routing Protocol (HSRP) notifications.
- **isdn**—Sends ISDN notifications. *Notification-option* arguments (see examples below) can be specified in combination with this keyword.
- **repeater**—Sends Ethernet repeater (hub) notifications. *Notification-option* arguments (see examples below) can be specified in combination with this keyword.
- **rsvp**—Sends Resource Reservation Protocol (RSVP) notifications.
- **rtr**—Sends Service Assurance Agent/Response Time Reporter (RTR) notifications.
- **snmp** [**authentication**]—Sends RFC 1157 SNMP notifications. Using the **authentication** keyword produces the same effect as not using it. Both the **snmp-server enable traps snmp** and the **snmp-server enable traps snmp authentication** forms of this command globally enable the following SNMP notifications (or, if you are using the **no** form of the command, disables such notifications): **authenticationFailure**, **linkUp**, **linkDown**, and **warmstart**.
- **syslog**—Sends system error message (syslog) notifications. You can specify the level of messages to be sent using the **logging history level** command.

notification-type (continued)	• **mpls ldp**—Sends notifications about status changes in LDP sessions. Note that this keyword is specified as **mpls ldp**. This syntax, which the CLI interprets as a two-word construct, has been implemented in this manner to maintain consistency with other MPLS commands. *Notification-option* arguments (below) can be specified in combination with this keyword. • **mpls traffic-eng**—Sends notifications about status changes in MPLS label distribution tunnels. This keyword is specified as **mpls traffic-eng**. This syntax, which the CLI interprets as a two-word construct, has been implemented in this manner to maintain consistency with other MPLS commands. *Notification-option* arguments (below) can be specified in combination with this keyword.
notification-option	(Optional) Defines the particular options associated with the specified *notification-type* that are to be enabled on the LSR. • **envmon** [**voltage** | **shutdown** | **supply** | **fan** | **temperature**] When you specify the **envmon** keyword, you can enable any one or all of the following environmental notifications in any combination: **voltage**, **shutdown**, **supply**, **fan**, or **temperature**. If you do not specify an argument with the **envmon** keyword, all types of system environmental notifications are enabled on the LSR. • **isdn** [**call-information** | **isdn u-interface**] When you specify the **isdn** keyword, you can use either the **call-information** keyword (to enable an SNMP ISDN call information option for the ISDN MIB subsystem) or the **isdn u-interface** keyword (to enable an SNMP ISDN U interface option for the ISDN U Interfaces MIB subsystem), or both. If you do not specify an argument with the **isdn** keyword, both types of isdn notifications are enabled on the LSR. • **repeater** [**health** | **reset**] When you specify the **repeater** keyword, you can use either the **health** argument or the **reset** keyword, or both (to enable the IETF Repeater Hub MIB [RFC 1516] notification). If you do not specify an argument with the **repeater** keyword, both types of notifications are enabled on the LSR. • **mpls ldp** [**session-up** | **session-down** | **pv-limit** | **threshold**] When you specify the **mpls ldp** keyword, you can use any one or all of the following arguments in any combination to indicate status changes in LDP sessions: **session-up**, **session-down**, **pv-limit**, or **threshold**. If you do not specify an argument with the **mpls ldp** keyword, all four types of LDP session notifications are enabled on the LSR. • **mpls traffic-eng** [**up** | **down** | **reroute**] When you specify the **mpls traffic-eng** keyword, you can use any one or all of the following arguments in any combination to enable the sending of notifications regarding status changes in MPLS label distribution tunnels: **up**, **down**, or **reroute**. If you do not specify an argument with the **mpls traffic-eng** keyword, all three types of tunnel notifications are enabled on the LSR.

Command Default

If you issue this command on an LSR without specifying any *notification-type* keywords, the default behavior of the LSR is to enable all notification types controlled by the command (some notification types cannot be controlled by means of this command).

Command Modes

Global configuration (config)

Command History

Release	Modification
11.1	This command was introduced.
11.3	The **snmp-server enable traps snmp authentication** form of this command was introduced to replace the **snmp-server trap-authentication** command.
12.0(17)ST	The **mpls traffic-eng** keyword was added to define a class or family of specific SNMP notifications for use with the *notification-type* and *notification-option* parameters of the **snmp-server enable traps** command.
12.0(21)ST	The **mpls ldp** keyword was added to define a class or family of specific SNMP notifications for use with the *notification-type* and *notification-option* parameters of the **snmp-server enable traps** command.
12.0(22)S	This command was integrated into Cisco IOS Release 12.0(22)S.
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.4(11)T	This command was integrated into Cisco IOS Release 12.4(11)T.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

To configure an LSR to send SNMP LDP notifications, you must issue at least one **snmp-server enable traps** command on the router.

To configure an LSR to send either notifications (traps) or informs to a designated network management station (NMS), you must issue the **snmp-server host** command on that device, using the keyword (**traps** or **informs**) that suits your purposes.

If you issue the **snmp-server enable traps** command without keywords, all SNMP notification types are enabled on the LSR. If you issue this command with specific keywords, only the notification types associated with those particular keywords are enabled on the LSR.

The **snmp-server enable traps** command is used in conjunction with the **snmp-server host** command. You use the latter command to specify the NMS host (or hosts) targeted as the recipient(s) of the SNMP notifications generated by SNMP-enabled LSRs in the network. To enable an LSR to send such notifications, you must issue at least one **snmp-server host** command on the LSR.

Examples

In the following example, the router is enabled to send all notifications to the host specified as myhost.cisco.com. The community string is defined as public.

```
Router(config)# snmp-server enable traps

Router(config)# snmp-server host myhost.cisco.com public
```

In the following example, the router is enabled to send Frame Relay and environmental monitor notifications to the host specified as myhost.cisco.com. The community string is defined as public:

```
Router(config)# snmp-server enable traps frame-relay

Router(config)# snmp-server enable traps envmon temperature

Router(config)# snmp-server host myhost.cisco.com public
```

In the following example, notifications are not sent to any host. BGP notifications are enabled for all hosts, but the only notifications enabled to be sent to a host are ISDN notifications (which are not enabled in this example).

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

Router(config)# snmp-server host host1 public isdn
```

In the following example, the router is enabled to send all inform requests to the host specified as myhost.cisco.com. The community string is defined as public.

```
Router(config)# snmp-server enable traps

Router(config)# snmp-server host myhost.cisco.com informs version 2c public
```

In the following example, HSRP MIB notifications are sent to the host specified as myhost.cisco.com. The community string is defined as public.

```
Router(config)# snmp-server enable hsrp

Router(config)# snmp-server host myhost.cisco.com traps version 2c public hsrp
```

Related Commands

Command	Description
snmp-server host	Specifies the intended recipient of an SNMP notification (that is, the designated NMS workstation in the network).

snmp-server group

To configure a new Simple Network Management Protocol (SNMP) group, use the **snmp-server group** command in global configuration mode. To remove a specified SNMP group, use the **no** form of this command.

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*]]

no snmp-server group *group-name* {**v1** | **v2c** | **v3** {**auth** | **noauth** | **priv**}} [**context** *context-name*]

Syntax Description

group-name	Name of the group.
v1	Specifies that the group is using the SNMPv1 security model. SNMPv1 is the least secure of the possible SNMP security models.
v2c	Specifies that the group is using the SNMPv2c security model. The SNMPv2c security model allows informs to be transmitted and supports 64-character strings.
v3	Specifies that the group is using the SNMPv3 security model. SMNPv3 is the most secure of the supported security models. It allows you to explicitly configure authentication characteristics.
auth	Specifies authentication of a packet without encrypting it.
noauth	Specifies no authentication of a packet.
priv	Specifies authentication of a packet with encryption.
context	(Optional) Specifies the SNMP context to associate with this SNMP group and its views.
context-name	(Optional) Context name.
read	(Optional) Specifies a read view for the SNMP group. This view enables you to view only the contents of the agent.
read-view	(Optional) String of a maximum of 64 characters that is the name of the view. The default is that the read-view is assumed to be every object belonging to the Internet object identifier (OID) space (1.3.6.1), unless the **read** option is used to override this state.
write	(Optional) Specifies a write view for the SNMP group. This view enables you to enter data and configure the contents of the agent.
write-view	(Optional) String of a maximum of 64 characters that is the name of the view. The default is that nothing is defined for the write view (that is, the null OID). You must configure write access.
notify	(Optional) Specifies a notify view for the SNMP group. This view enables you to specify a notify, inform, or trap.

notify-view	(Optional) String of a maximum of 64 characters that is the name of the view. By default, nothing is defined for the notify view (that is, the null OID) until the **snmp-server host** command is configured. If a view is specified in the **snmp-server group** command, any notifications in that view that are generated will be sent to all users associated with the group (provided a SNMP server host configuration exists for the user). Cisco recommends that you let the software autogenerate the notify view. See the "Configuring Notify Views" section in this document.
access	(Optional) Specifies a standard access control list (ACL) to associate with the group.
ipv6	(Optional) Specifies an IPv6 named access list. If both IPv6 and IPv4 access lists are indicated, the IPv6 named access list must appear first in the list.
named-access-list	(Optional) Name of the IPv6 access list.
[*acl-number* | *acl-name*]	(Optional) The *acl-number* argument is an integer from 1 to 99 that identifies a previously configured standard access list. The *acl-name* argument is a string of a maximum of 64 characters that is the name of a previously configured standard access list.

Command Default

No SNMP server groups are configured.

Command Modes

Global configuration (config)

Command History

Release	Modification
11.(3)T	This command was introduced.
12.0(23)S	The **context** *context-name* keyword and argument pair was added.
12.3(2)T	The **context** *context-name* keyword and argument pair was integrated into Cisco IOS Release 12.3(2)T, and support for standard named access lists (acl-name) was added.
12.0(27)S	The **ipv6** *named-access-list* keyword and argument pair was added.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.3(14)T	The **ipv6** *named-access-list* keyword and argument pair was integrated into Cisco IOS Release 12.3(14)T.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	This command was integrated into Cisco IOS Release 12.2(31)SB2.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

When a community string is configured internally, two groups with the name public are autogenerated, one for the v1 security model and the other for the v2c security model. Similarly, deleting a community string will delete a v1 group with the name public and a v2c group with the name public.

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 Message Digest 5 (MD5) password, see the documentation of the **snmp-server user** command.

Configuring Notify Views

The *notify-view* option is available for two reasons:

- If a group has a notify view that is set using SNMP, you may need to change the notify view.
- The **snmp-server host** command may have been configured before the **snmp-server group** command. In this case, you must either reconfigure the **snmp-server host** command, or specify the appropriate notify view.

Specifying a notify view when configuring an SNMP group is not recommended, for the following reasons:

- The **snmp-server host** command autogenerates a notify view for the user, and then adds it to the group associated with that user.
- Modifying the group's notify view will affect all users associated with that group.

Instead of specifying the notify view for a group as part of the **snmp-server group** command, use the following commands in the order specified:

1. **snmp-server user**—Configures an SNMP user.
2. **snmp-server group**—Configures an SNMP group, without adding a notify view.
3. **snmp-server host**—Autogenerates the notify view by specifying the recipient of a trap operation.

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 users of other VPNs on the same networking device.

Use this command with the **context** *context-name* keyword and argument to associate a read, write, or notify SNMP view with an SNMP context.

Examples

Create an SNMP Group

The following example shows how to create the SNMP server group "public," allowing read-only access for all objects to members of the standard named access list "lmnop":

```
Router(config)# snmp-server group public v2c access lmnop
```

Remove an SNMP Server Group

The following example shows how to remove the SNMP server group "public" from the configuration:

```
Router(config)# no snmp-server group public v2c
```

Associate an SNMP Server Group with Specified Views

The following example shows SNMP context "A" associated with the views in SNMPv2c group "GROUP1":

```
Router(config)# snmp-server context A
Router(config)# snmp mib community commA
Router(config)# snmp mib community-map commA context A target-list commAVpn
Router(config)# snmp-server group GROUP1 v2c context A read viewA write viewA notify viewB
```

Related Commands

Command	Description
show snmp group	Displays the names of groups on the router and the security model, the status of the different views, and the storage type of each group.
snmp mib community-map	Associates a SNMP community with an SNMP context, engine ID, security name, or VPN target list.
snmp-server host	Specifies the recipient of a SNMP notification operation.
snmp-server user	Configures a new user to a SNMP group.

snmp-server host

To specify the recipient of a Simple Network Management Protocol (SNMP) notification operation, use the **snmp-server host** command in global configuration mode. To remove the specified host from the configuration, use the **no** form of this command.

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*]

no 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*]

Syntax Description

hostname | *ip-address*	Name, IP address, or IPv6 address of the SNMP notification host. The *ip-address* can be an IP or IPv6 address. The SNMP notification host is typically a network management station (NMS or SNMP manager). This host is the recipient of the SNMP traps or informs.
vrf	(Optional) Specifies that a Virtual Private Network (VPN) routing and forwarding (VRF) instance should be used to send SNMP notifications.
vrf-name	(Optional) VPN VRF instance used to send SNMP notifications.
traps	(Optional) Specifies that notifications should be sent as traps. This is the default.
informs	(Optional) Specifies that notifications should be sent as informs.
version	(Optional) Version of the SNMP used to send the traps. The default is 1. If you use the **version** keyword, one of the following keywords must be specified: • **1**—SNMPv1. This option is not available with informs. • **2c**—SNMPv2C. • **3**—SNMPv3. The most secure model because it allows packet encryption with the **priv** keyword. The default is noauth. One of the following three optional security level keywords can follow the **3** keyword: – **auth**—Enables Message Digest 5 (MD5) and Secure Hash Algorithm (SHA) packet authentication. – **noauth**—Specifies that the noAuthNoPriv security level applies to this host. This is the default security level for SNMPv3. – **priv**—Enables Data Encryption Standard (DES) packet encryption (also called "privacy").
community-string	Password-like community string is sent with the notification operation. **Note** You can set this string using the **snmp-server host** command by itself, but Cisco recommends that you define the string using the **snmp-server community** command prior to using the **snmp-server host** command. **Note** The sign (@) is used for delimiting the context information.

udp-port	(Optional) Specifies that SNMP notifications or informs are to be sent to an NMS host.
port	(Optional) UDP port number of the NMS host. The default is 162.
notification-type	(Optional) Type of notification to be sent to the host. If no type is specified, all available notifications are sent. The notification type can be one or more of the following keywords: • **bgp**—Sends Border Gateway Protocol (BGP) state change notifications. • **calltracker**—Sends Call Tracker call-start/call-end notifications. • **cef** — Sends Cisco Express Forwarding-related notifications. • **config**—Sends configuration change notifications. • **cpu**—Sends CPU-related notifications. • **director**—Sends DistributedDirector-related notifications. • **dspu**—Sends downstream physical unit (DSPU) notifications. • **eigrp**—Sends Enhanced Interior Gateway Routing Protocol (EIGRP) stuck-in-active (SIA) and neighbor authentication failure notifications. • **entity**—Sends Entity MIB modification notifications. • **envmon**—Sends Cisco enterprise-specific environmental monitor notifications when an environmental threshold is exceeded. • **flash**—Sends flash media insertion and removal notifications. • **frame-relay**—Sends Frame Relay notifications. • **hsrp**—Sends Hot Standby Routing Protocol (HSRP) notifications. • **iplocalpool**—Sends IP local pool notifications. • **ipmobile**—Sends Mobile IP notifications. • **ipsec**—Sends IP Security (IPsec) notifications. • **isdn**—Sends ISDN notifications. • **l2tun-pseudowire-status**—Sends pseudowire state change notifications. • **l2tun-session**—Sends Layer 2 tunneling session notifications. • **llc2**—Sends Logical Link Control, type 2 (LLC2) notifications. • **memory**—Sends memory pool and memory buffer pool notifications. • **mpls-ldp**—Sends Multiprotocol Label Switching (MPLS) Label Distribution Protocol (LDP) notifications indicating status changes in LDP sessions. • **mpls-traffic-eng**—Sends MPLS traffic engineering notifications indicating changes in the status of MPLS traffic engineering tunnels. • **mpls-vpn**—Sends MPLS VPN notifications. • **ospf**—Sends Open Shortest Path First (OSPF) sham-link notifications. • **pim**—Sends Protocol Independent Multicast (PIM) notifications. • **repeater**—Sends standard repeater (hub) notifications.

- **rsrb**—Sends remote source-route bridging (RSRB) notifications.
- **rsvp**—Sends Resource Reservation Protocol (RSVP) notifications.
- **rtr**—Sends Response Time Reporter (RTR) notifications.
- **sdlc**—Sends Synchronous Data Link Control (SDLC) notifications.
- **sdllc**—Sends SDLC Logical Link Control (SDLLC) notifications.
- **snmp**—Sends any enabled RFC 1157 SNMP linkUp, linkDown, authenticationFailure, warmStart, and coldStart notifications.

Note To enable RFC 2233 compliant link up/down notifications, you should use the **snmp server link trap** command.

- **srp**—Sends Spatial Reuse Protocol (SRP) notifications.
- **stun**—Sends serial tunnel (STUN) notifications.
- **syslog**—Sends error message notifications (Cisco Syslog MIB). Specify the level of messages to be sent with the **logging history level** command.
- **tty**—Sends Cisco enterprise-specific notifications when a TCP connection closes.
- **voice**—Sends SNMP poor quality of voice traps, when used with the **snmp enable peer-trap poor qov** command.
- **vrrp**—Sends Virtual Router Redundancy Protocol (VRRP) notifications.
- **vsimaster**—Sends Virtual Switch Interface (VSI) Master notifications.
- **x25**—Sends X.25 event notifications.

Command Default

This command is disabled by default. No notifications are sent.

Command Modes

Global configuration (config)

Command History

Release	Modification
10.0	This command was introduced.
Cisco IOS Release 12 Mainline/T Train	
12.0(3)T	• The **version 3** [**auth** | **noauth** | **priv**] syntax was added as part of the SNMPv3 Support feature. • The **hsrp** notification-type keyword was added. • The **voice** notification-type keyword was added.
12.1(3)T	The **calltracker** notification-type keyword was added for the Cisco AS5300 and AS5800 platforms.

Release	Modification
12.2(2)T	• The **vrf** *vrf-name* keyword/argument combination was added. • The **ipmobile** notification-type keyword was added. • Support for the **vsimaster** notification-type keyword was added for the Cisco 7200 and Cisco 7500 series.
12.2(4)T	• The **pim** notification-type keyword was added. • The **ipsec** notification-type keyword was added.
12.2(8)T	• The **mpls-traffic-eng** notification-type keyword was added. • The **director** notification-type keyword was added.
12.2(13)T	• The **srp** notification-type keyword was added. • The **mpls-ldp** notification-type keyword was added.
12.3(2)T	• The **flash** notification-type keyword was added. • The **l2tun-session** notification-type keyword was added.
12.3(4)T	• The **cpu** notification-type keyword was added. • The **memory** notification-type keyword was added. • The **ospf** notification-type keyword was added.
12.3(8)T	The **iplocalpool** notification-type keyword was added for the Cisco 7200 and 7301 series routers.
12.3(11)T	The **vrrp** keyword was added.
12.3(14)T	• Support for SNMP over IPv6 transport was integrated into Cisco IOS Release 12.3(14)T. Either an IP or IPv6 Internet address can be specified as the *hostname* argument. • The **eigrp** notification-type keyword was added.
Cisco IOS Release 12.0S	
12.0(17)ST	The **mpls-traffic-eng** notification-type keyword was added.
12.0(21)ST	The **mpls-ldp** notification-type keyword was added.
12.0(22)S	• All features in the Cisco IOS Release 12.0ST train were integrated into Cisco IOS Release 12.0(22)S. • The **mpls-vpn** notification-type keyword was added.
12.0(23)S	The **l2tun-session** notification-type keyword was added.
12.0(26)S	The **memory** notification-type keyword was added.
12.0(27)S	• Support for SNMP over IPv6 transport was added. Either an IP or IPv6 Internet address can be specified as the *hostname* argument. • The **vrf** *vrf-name* keyword argument pair was integrated into Cisco IOS Release 12.0(27)S to support multiple Lightweight Directory Protocol (LDP) contexts for VPNs.
12.0(31)S	The **l2tun-pseudowire-status** notification-type keyword was added.
Release 12.2S	
12.2(18)S	This command was integrated into Cisco IOS Release 12.2(18)S.

Release	Modification
12.2(25)S	• The **cpu** notification-type keyword was added. • The **memory** notification-type keyword was added.
12.2(28)SB	This command was integrated into Cisco IOS Release 12.2(28)SB.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(31)SB2	The **cef** notification-type keyword was added.
12.2(31)SB3	This command was implemented on the Cisco 10000 series.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

If you enter this command with no keywords, the default is to send all trap types to the host. No informs will be sent to the host.

The **no snmp-server host** command with no keywords disables traps, but not informs, to the host. To disable informs, use the **no snmp-server host informs** command.

Note If the community-string is not defined using the **snmp-server community** command prior to using this command, the default form of the **snmp-server community** command will automatically be inserted into the configuration. The password (community-string) used for this automatic configuration of the **snmp-server community** will be the same as specified in the **snmp-server host** command. This automatic command insertion and use of passwords is the default behavior for Cisco IOS Release 12.0(3) and later releases.

SNMP notifications can be sent as traps or inform requests. Traps are unreliable because the receiver does not send acknowledgments when it receives traps. The sender cannot determine if the traps were received. However, a SNMP entity that receives an inform request acknowledges the message with a SNMP response protocol data unit (PDU). If the sender never receives the response, the inform request 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 request 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 retried several times. The retries increase traffic and contribute to a higher 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 with no 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 kind of notification (trap or inform), 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 will replace 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 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.

The **vrf** keyword allows you to specify the notifications being sent to a specified IP address over a specific VRF. The VRF defines a VPN membership of a customer so data is stored using the VPN.

Notification-Type Keywords

The *notification-type* keywords used in the **snmp-server host** command do not always match the keywords used in the corresponding **snmp-server enable traps** command. For example, the notification keyword applicable to Multiprotocol Label Switching Protocol (MPLS) traffic engineering tunnels is specified as **mpls-traffic-eng** (containing two hyphens and no intervening spaces). The corresponding parameter in the **snmp-server enable traps** command is specified as **mpls-traffic-eng** (containing an intervening space and a hyphen).

This syntax difference is necessary to ensure that the command-line interface (CLI) interprets the *notification-type* keyword of the **snmp-server host** command as a unified, single-word construct, which preserves the capability of the **snmp-server host** command to accept multiple *notification-type* keywords in the command line. The **snmp-server enable traps** commands, however, often use two-word constructs to provide hierarchical configuration options and to maintain consistency with the command syntax of related commands. Table 12 maps some examples of **snmp-server enable traps** commands to the keywords used in the **snmp-server host** command.

Table 12 Notification Keywords and Corresponding SNMP Enable Traps Commands

SNMP Enable Traps Command	SNMP Host Command Keyword
snmp-server enable traps l2tun session	**l2tun-session**
snmp-server enable traps mpls ldp	**mpls-ldp**
snmp-server enable traps mpls traffic-eng[1]	**mpls-traffic-eng**
snmp-server enable traps mpls vpn	**mpls-vpn**

1. See the *Cisco IOS Multiprotocol Label Switching Command Reference* for documentation of this command.

Examples

If you want to configure a unique SNMP community string for traps but prevent SNMP polling access with this string, the configuration should include an access list. The following example shows how to name a community string comaccess and number an access list 10:

```
Router(config)# snmp-server community comaccess ro 10
Router(config)# snmp-server host 172.20.2.160 comaccess
Router(config)# access-list 10 deny any
```

Note The sign (@) is used as a delimiter between the community string and the context in which it is used. For example, specific VLAN information in BRIDGE-MIB may be polled using community@VLAN_ID (for example, public@100) where 100 is the VLAN number.

The following example shows how to send RFC 1157 SNMP traps to a host specified named myhost.cisco.com. Other traps are enabled, but only SNMP traps are sent because only **snmp** is specified in the **snmp-server host** command. The community string is defined as comaccess.

```
Router(config)# snmp-server enable traps
Router(config)# snmp-server host myhost.cisco.com comaccess snmp
```

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

```
Router(config)# snmp-server enable traps snmp
Router(config)# snmp-server enable traps envmon
Router(config)# 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 myhost.cisco.com using the community string public:

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

The following example will not send traps to any host. The BGP traps are enabled for all hosts, but only the ISDN traps are enabled to be sent to a host. The community string is defined as public.

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

The following example shows how to enable the router to send all inform requests to the host myhost.cisco.com using the community string public:

```
Router(config)# snmp-server enable traps
Router(config)# snmp-server host myhost.cisco.com informs version 2c public
```

The following example shows how to send HSRP MIB informs to the host specified by the name myhost.cisco.com. The community string is defined as public.

```
Router(config)# snmp-server enable traps hsrp
Router(config)# snmp-server host myhost.cisco.com informs version 2c public hsrp
```

The following example shows how to send all SNMP notifications to company.com over the VRF named trap-vrf using the community string public:

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

The following example shows how to configure an IPv6 SNMP notification server with the IPv6 address 2001:0DB8:0000:ABCD:1 using the community string public:

```
Router(config)# snmp-server host 2001:0DB8:0000:ABCD:1 version 2c public udp-port 2012
```

The following example shows how to specify VRRP as the protocol using the community string public:

```
Router(config)# snmp-server enable traps vrrp
Router(config)# snmp-server host myhost.cisco.com traps version 2c public vrrp
```

The following example shows how to send all Cisco Express Forwarding informs to the notification receiver with the IP address 10.56.125.47 using the community string public:

```
Router(config)# snmp-server enable traps cef
Router(config)# snmp-server host 10.56.125.47 informs version 2c public cef
```

Related Commands

Command	Description
snmp-server enable peer-trap poor qov	Enables poor quality of voice notifications for applicable calls associated with a specific voice dial peer.
snmp-server enable traps	Enables SNMP notifications (traps and informs).
snmp-server informs	Specifies inform request options.
snmp-server link trap	Enables linkUp/linkDown SNMP traps, which are compliant with RFC 2233.
snmp-server trap-source	Specifies the interface (and hence the corresponding IP address) from which a SNMP trap should originate.
snmp-server trap-timeout	Defines how often to try resending trap messages on the retransmission queue.

snmp-server trap authentication vrf

To enable Virtual Private Network (VPN) routing and forwarding (VRF) instance context authentication notifications for Simple Network Management Protocol (SNMP), use the **snmp-server trap authentication vrf** command in global configuration mode. To suppress authentication notifications for SNMP packets dropped due specifically to VRF context mismatches while keeping all other SNMP authentication notifications enabled, use the **no** form of this command.

snmp-server trap authentication vrf

no snmp-server trap authentication vrf

Syntax Description

This command has no arguments or keywords.

Command Default

No VRF-specific authentication notifications are enabled when SNMP authentication notifications are not enabled.

Command Modes

Global configuration (config)

Command History

Release	Modification
12.0(23)S	This command was introduced.
12.3(2)T	This command was integrated into Cisco IOS Release 12.3(2)T.
12.2(25)S	This command was integrated into Cisco IOS Release 12.2(25)S.
12.2(33)SRA	This command was integrated into Cisco IOS Release 12.2(33)SRA.
12.2(33)SXH	This command was integrated into Cisco IOS Release 12.2(33)SXH.

Usage Guidelines

The **snmp-server enable traps snmp authentication** command controls SNMP authentication traps and the **no** form of this command disables all SNMP authentication failure notifications. The **snmp-server trap authentication vrf** command provides more granular control of these notifications.

With context-based MIB access, SNMP requests on each VRF are tied to a specific context. This context is used for access control. If SNMP contexts are configured for VPNs, any SNMP request not matching the configured context will generate an SNMP authentication failure notification.The **no snmp-server trap authentication vrf** command allows you to suppress the authentication failure notifications that are specific to these VRF contexts, while keeping all other SNMP authentication failure notifications enabled.

The **no snmp-server trap authentication vrf** command has no effect if the **snmp-server enable traps snmp authentication** command has not been configured.

Examples

The following example shows how to enable a router to send SNMP authentication traps to host myhost.cisco.com using the community string public while disabling all VRF authentication traps:

```
Router(config)# snmp-server enable traps snmp authentication
Router(config)# no snmp-server trap authentication vrf
Router(config)# snmp-server host myhost.cisco.com public
```

Related Commands

Command	Description
snmp-server enable traps snmp	Enables the sending of RFC 1157 SNMP notifications.
snmp-server host	Specifies the recipient of an SNMP notification operation.

Glossary

ATM—Asynchronous Transfer Mode. The international standard for cell relay in which multiple service types (such as voice, video, or data) are conveyed in fixed-length (53-byte) cells. Fixed-length cells allow cell processing to occur in hardware, thereby reducing transit delays. ATM is designed to take advantage of high-speed transmission media, such as E3, SONET, and T3.

downstream-on-demand distribution—A label distribution method in which a downstream label switch router (LSR) sends a binding upstream only if the upstream LSR requests it.

downstream unsolicited distribution—A label distribution method in which labels are dispersed if a downstream label switch router (LSR) needs to establish a new binding with its neighboring upstream LSR. For example, an edge LSR might enable a new interface with another subnet. The LSR then announces to the upstream router a binding to reach this network.

informs—A type of notification message that is more reliable than a conventional trap notification message, because the informs message notification requires acknowledgment, but a trap notification does not.

label—A short, fixed-length data identifier that tells switching nodes how to forward data (packets or cells).

label distribution—The techniques and processes that are used by label switch routers (LSRs) to exchange label binding information for supporting hop-by-hop forwarding along normally routed paths.

LDP—Label Distribution Protocol. The protocol that supports MPLS hop-by-hop forwarding and the distribution of bindings between labels and network prefixes. The Cisco proprietary version of this protocol is the Tag Distribution Protocol (TDP).

LSP—label-switched path. A configured connection between two label switch routers (LSRs) in which label-switching techniques are used for packet forwarding; also a specific path through an MPLS network.

LSR—label switch router. A Multiprotocol Label Switching (MPLS) node that can forward native Layer 3 packets. The LSR forwards a packet based on the value of a label attached to the packet.

MIB—Management Information Base. A database of network management information that is used and maintained by a network management protocol such as Simple Network Management Protocol (SNMP). The value of a MIB object can be changed or retrieved by the use of SNMP commands, usually through a network management system. MIB objects are organized in a tree structure that includes public (standard) and private (proprietary) branches.

MPLS—Multiprotocol Label Switching. A switching method for the forwarding of IP traffic through the use of a label. This label instructs the routers and the switches in the network where to forward the packets based on preestablished IP routing information.

MPLS label distribution—A constraint-based routing algorithm for routing label-switched path (LSP) tunnels.

NMS—network management station. A powerful, well-equipped computer (typically an engineering workstation) that is used by a network administrator to communicate with other devices in the network. An NMS is typically used to manage network resources, gather statistics, and perform a variety of network administration and configuration tasks. In the context of SNMP, an NMS is a device that performs SNMP queries to the SNMP agent of a managed device to retrieve or modify information.

notification—A message sent by a Simple Network Management Protocol (SNMP) agent to a network management station, console, or terminal to indicate that a significant network event has occurred. *See also* trap.

RSVP—Resource Reservation Protocol. A protocol that supports the reservation of resources across an IP network. Applications running on IP end systems can use RSVP to indicate to other nodes the nature of the packet streams they want to receive by specifying such items as bandwidth, jitter, and maximum burst.

RTR—Response Time Reporter. A tool that allows you to monitor network performance, network resources, and applications by measuring response times and availability.

SNMP—Simple Network Management Protocol. A network management protocol used almost exclusively in TCP/IP networks. SNMP enables a user to monitor and control network devices, manage configurations, collect statistics, monitor performance, and ensure network 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 as well as 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.

TDP—Tag Distribution Protocol. A standard protocol used by MPLS-enabled routers to negotiate the tags (addresses) used for forwarding packets. *See also* LDP.

TLV—Type-Length-Value. A mechanism used by several routing protocols to carry a variety of attributes. Cisco Discovery Protocol (CDP), Label Discovery Protocol (LDP), and Border Gateway Protocol (BGP) are examples of protocols that use TLVs. BGP uses TLVs to carry attributes such as Network Layer Reachability Information (NLRI), Multiple Exit Discriminator (MED), and local preference.

trap—A message sent by an SNMP agent to a network management station, console, or terminal to indicate that a significant network event has occurred. Traps (notifications) are less reliable than inform requests, because the receiver of the trap does not send an acknowledgment of receipt; furthermore, the sender of the trap cannot determine if the trap was received. *See also* notification.

VCC—virtual channel connection. A logical circuit, made up of virtual channel links (VCLs), that carries data between two endpoints in an ATM network. Sometimes called a *virtual circuit connection*.

VCI—virtual channel identifier. A 16-bit field in the header of an ATM cell. The VCI, together with the virtual path identifier (VPI), is used to identify the next network virtual channel link (VCL) as the cell passes through a series of ATM switches on its way to its final destination.

VCL—virtual channel link. The logical connection that exists between two adjacent switches in an ATM network.

VPI—virtual path identifier. An 8-bit field in the header of an ATM cell. The VPI, together with the virtual channel identifier (VCI), is used to identify the next network virtual channel link (VCL) as the cell passes through a series of ATM switches on its way to its final destination.

VPN—Virtual Private Network. A network that enables IP traffic to use tunneling to travel securely over a public TCP/IP network.

VRF—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.