

MPLS Traffic Engineering MIB

The MPLS Traffic Engineering MIB enables Simple Network Management Protocol (SNMP) agent support in Cisco software for Multiprotocol Label Switching (MPLS) traffic engineering (TE) management, as implemented in the MPLS Traffic Engineering MIB (MPLS TE MIB). The SNMP agent code operating with the MPLS TE MIB enables a standardized, SNMP-based approach to be used in managing the MPLS TE features in Cisco software.

- Restrictions for the MPLS Traffic Engineering MIB, on page 1
- Information About the MPLS Traffic Engineering MIB, on page 1
- How to Configure the MPLS Traffic Engineering MIB, on page 9
- Configuration Examples for the MPLS Traffic Engineering MIB, on page 11
- Additional References, on page 12
- Feature Information for the MPLS Traffic Engineering MIB, on page 13
- Glossary, on page 13

Restrictions for the MPLS Traffic Engineering MIB

- Supports read-only (RO) permission for MIB objects.
- Contains no configuration support by means of SET functions, except for the mplsTunnelTrapEnable object (which has been made writable). Accordingly, the MPLS TE MIB contains indexing support for the Interfaces MIB.
- Supports only SNMP GET, GETNEXT, and GETBULK retrieval functions, except in the case of the mplsTunnelTrapEnable object (which has been made writable by means of SET functions).
- Contains no support for Guaranteed Bandwidth Traffic Engineering (GBTE) or Auto Bandwidth features.

Information About the MPLS Traffic Engineering MIB

MPLS Traffic Engineering MIB Cisco Implementation

The MPLS TE MIB is based on the Internet Engineering Task Force (IETF) draft MIB entitled *draft-ietf-mpls-te-mib-05.txt* which includes objects describing features that support MPLS TE.

Slight differences between the IETF draft MIB and the implementation of the TE capabilities within Cisco software require some minor translations between the MPLS TE MIB and the internal data structures of Cisco software. These translations are made by the SNMP agent code that is installed and operating on various hosts within the network. This SNMP agent code, running in the background as a low priority process, provides a management interface to Cisco software.

The SNMP objects defined in the MPLS TE MIB can be displayed by any standard SNMP utility. All MPLS TE MIB objects are based on the IETF draft MIB; thus, no specific Cisco SNMP application is required to support the functions and operations pertaining to the MPLS TE MIB.

MPLS Traffic Engineering Overview

MPLS TE capabilities in Cisco software enable an MPLS backbone to replicate and expand upon the TE capabilities of Layer 2 ATM and Frame Relay networks.

TE capabilities are essential to effective management of service provider and Internet service provider (ISP) backbones. Such backbones must support high transmission capacities, and the networks incorporating backbones must be extremely resilient to link or node failures.

The MPLS TE facilities built into Cisco software provide a feature-rich, integrated approach to managing the large volumes of traffic that typically flow through WANs. The MPLS TE facilities are integrated into Layer 3 network services, thereby optimizing the routing of IP traffic in the face of constraints imposed by existing backbone transmission capacities and network topologies.

Capabilities Supported by the MPLS Traffic Engineering MIB

- The ability to generate and queue notification messages that signal changes in the operational status of MPLS TE tunnels.
- Extensions to existing SNMP commands that provide the ability to enable, disable, and configure notification messages for MPLS TE tunnels.
- The ability to specify the name or the IP address of a network management station (NMS) in the operating environment to which notification messages are to be sent.
- The ability to write notification configurations into nonvolatile memory.

Notification Generation Events

When MPLS TE notifications are enabled (see the **snmp-server enable traps mpls** command), notification messages relating to specific events within Cisco software are generated and sent to a specified NMS in the network.

For example, an mplsTunnelUp notification is sent to an NMS when an MPLS TE tunnel is configured and the tunnel transitions from an operationally “down” state to an “up” state.

Conversely, an mplsTunnelDown notification is generated and sent to an NMS when an MPLS TE tunnel transitions from an operationally “up” state to a “down” state.

An mplstunnelRerouted notification is sent to the NMS under the following conditions:

- The signaling path of an existing MPLS TE tunnel fails for some reason and a new path option is signaled and placed into effect (that is, the tunnel is rerouted).

- The signaling path of an existing MPLS TE tunnel is fully operational, but a better path option can be signaled and placed into effect (that is, the tunnel can be reoptimized). This reoptimazation can be triggered by:
  - A timer
  - The issuance of an **mpls traffic-eng reoptimize** command
  - A configuration change that requires the resignaling of a tunnel

The mplsTunnelReoptimized notification is not generated when an MPLS traffic engineering tunnel is reoptimized. However, an mplsTunnelReroute notification is generated. Thus, at the NMS, you cannot distinguish between a tunnel reoptimization event and tunnel reroute event.

Path options are configurable parameters that you can use to specify the order of priority for establishing a new tunnel path. For example, you can create a tunnel head configuration and define any one of many path options numbered 1 through n, with “1” being the highest priority option and “n” being an unlimited number of lower priority path options. Thus, there is no limit to the number of path options that you can specify in this manner.

Notification Implementation

When an MPLS TE tunnel interface (or any other device interface, such as an FastEthernet or Packet over SONET (POS) interface) transitions between an up and down state, an Interfaces MIB (ifMIB) link notification is generated. When such a notification occurs in an MPLS TE MIB environment, the interface is checked by software to determine if the notification is associated with an MPLS TE tunnel. If so, the interfaces MIB link notification is interlinked with the appropriate mplsTunnelUp or mplsTunnelDown notification to provide notification to the NMS regarding the operational event occurring on the tunnel interface. Hence, the generation of an Interfaces MIB link notification pertaining to an MPLS traffic engineering tunnel interface begets an appropriate mplsTunnelUp or mplsTunnelDown notification that is transmitted to the specified NMS.

An mplsTunnelRerouted notification is generated whenever the signaling path for an MPLS TE tunnel changes. However, software intelligence in the MPLS TE MIB prevents the reroute notification from being sent to the NMS when a TE tunnel transitions between an up or down state during an administrative or operational status check of the tunnel. Either an up or down notification or a reroute notification can be sent in this instance, but not both. This action prevents unnecessary traffic on the network.

Benefits of the MPLS Traffic Engineering MIB

- Provides a standards-based SNMP interface for retrieving information about MPLS TE.
- Provides information about the traffic flows on MPLS TE tunnels.
- Presents MPLS TE tunnel routes, including the configured route, the Interior Gateway Protocol (IGP) calculated route, and the actual route traversed.
- Provides information, in conjunction with the Interfaces MIB, about how a tunnel was rerouted in the event of a link failure.
- Provides information about the configured resources used for an MPLS TE tunnel.
- Supports the generation and queueing of notifications that call attention to major changes in the operational status of MPLS TE tunnels;
- Forwards notification messages to a designated NMS for evaluation or action by network administrators.

MPLS Traffic Engineering MIB Layer Structure

The SNMP agent code supporting the MPLS TE MIB follows the existing model for such code in Cisco software and is, in part, generated by the Cisco tool set, based on the MIB source code.

The SNMP agent code, which has a layered structure similar to that of the MIB support code in Cisco software, consists of four layers:

- Platform independent layer--This layer is generated primarily by the Cisco MIB development tool set and incorporates platform and implementation independent functions. The Cisco MIB development tool set creates a standard set of files associated with a MIB.
- Application interface layer--The functions, names, and template code for MIB objects in this layer are also generated by the Cisco MIB development tool set.
- Application specific layer--This layer provides an interface between the application interface layer and the application program interface (API) and data structures layer and performs tasks needed to retrieve required information from Cisco software, such as searching through data structures.
- API and data structures layer--This layer contains the data structures or APIs within Cisco software that are retrieved or called in order to set or retrieve SNMP management information.

Features and Technologies Related to the MPLS Traffic Engineering MIB

The MPLS TE MIB feature is used with the following features and technologies:

- Standards-based SNMP network management application
- MPLS
- MPLS TE

Supported Objects in the MPLS Traffic Engineering MIB

The MPLS TE MIB contains numerous tables and object definitions that provide read-only SNMP management support for the MPLS TE features in Cisco software. The MPLS TE MIB conforms to Abstract Syntax Notation One (ASN.1), thus reflecting an idealized MPLS TE database.

Using any standard SNMP network management application, you can retrieve and display information from the MPLS TE MIB by using GET operations; similarly, you can traverse information in the MIB database for display by using GETNEXT operations.

The MPLS TE MIB tables and objects supported in Cisco software follow. Important MIB tables (those highlighted in bold type) are described briefly in accompanying text.

- mplsTunnelConfigured--Total number of tunnel configurations that are defined on this node.
- mplsTunnelActive--Total number of label switched paths (LSPs) that are defined on this node.
- mplsTunnelTEDistProto--The IGP distribution protocol in use.
- mplsTunnelMaxHops--The maximum number of hops any given tunnel can utilize.
- mplsTunnelIndexNext--Unsupported; set to 0.

- mplsTunnelTable--Entries in this table with an instance of 0 and a source address of 0 represent tunnel head configurations. All other entries in this table represent instances of LSPs, both signaled and standby. If a tunnel instance is signaled, its operating status (operStatus) is set to “up” (1) and its instance corresponds to an active LSP.

Tunnel configurations exist only on the tunnel head where the tunnel interface is defined. LSPs traverse the network and involve tunnel heads, tunnel midpoints, and tunnel tails.

Pointers in the tunnel table refer to corresponding entries in other MIB tables. By using these pointers, you can find an entry in the mplsTunnelTable and follow a pointer to other tables for additional information. The pointers are the following: mplsTunnelResourcePointer, mplsTunnelHopTableIndex, mplsTunnelARHopTableIndex, and mplsTunnelCHopTableIndex.

The tunnel table is indexed by tunnel ID, tunnel instance, tunnel source address, and tunnel destination address. The description of each entry has an alphabetic suffix (a) for tunnel head configurations only, (b) for LSPs only, or (c) for both tunnel head configurations and LSPs , if appropriate, to indicate the applicability of the entry.

Following is a list and description of each entry.

- - mplsTunnelIndex--Same as tunnel ID (c).
  - mplsTunnelInstance--Tunnel instance of the LSP; 0 for head configurations (b).
  - mplsTunnelIngressLSRId--Source IP address of the LSP; 0 for head configurations (b).
  - mplsTunnelEgressLSRId--Destination IP address of the tunnel (c).
  - mplsTunnelName--Command name for the tunnel interfaces (a).
  - mplsTunnelDescr--Descriptive name for tunnel configurations and LSPs (c).
  - mplsTunnelIsIf--Indicator of whether the entry represents an interface (c).
  - mplsTunnelIfIndex--Index of the tunnel interface within the ifMIB (a).
  - mplsTunnelXCPointer--(For midpoints only - no tails) Pointer for the LSP within the mplsXCTable of the MPLS LSR MIB (b).
  - mplsTunnelSignallingProto--Signaling protocol used by tunnels (c).
  - mplsTunnelSetupPrio--Setup priority of the tunnel (c).
  - mplsTunnelHoldingPrio--Holding priority of the tunnel (c).
  - mplsTunnelSessionAttributes--Session attributes (c).
  - mplsTunnelOwner--Tunnel owner (c).
  - mplsTunnelLocalProtectInUse-- Not supported on midpoint node (c).
  - mplsTunnelResourcePointer--Pointer into the Resource Table (b).
  - mplsTunnelInstancePriority--Not implemented (b).
  - mplsTunnelHopTableIndex--Index into the Hop Table (a).
  - mplsTunnelARHopTableIndex--Index into the AR Hop Table (b).
  - mplsTunnelCHopTableIndex--Index into the C Hop Table (b).
  - mplsTunnelPrimaryTimeUp--Amount of time, in seconds, that the current path has been up (a).
  - mplsTunnelPathChanges--Number of times a tunnel has been resignalled (a).
  - mplsTunnelLastPathChange--Amount of time, in seconds, since the last path resignaling occurred (a).
  - mplsTunnelCreationTime--Time stamp when the tunnel was created (a).
  - mplsTunnelStateTransitions--Number of times the tunnel has changed state (a).
  - mplsTunnelIncludeAnyAffinity--Not implemented (a).
  - mplsTunnelIncludeAllAffinity--Attribute bits that must be set for the tunnel to traverse a link (a).

- mplsTunnelExcludeAllAffinity--Attribute bits that must *not* be set for the tunnel to traverse a link (a).
- mplsTunnelPathInUse--Path option number being used for the tunnel's path. If no path option is active, this object will be 0 (a).
- mplsTunnelRole--Role of the tunnel on the router; that is, head, midpoint, or tail (c).
- mplsTunneltotalUptime--Amount of time, in seconds, that the tunnel has been operationally up (a).
- mplsTunnelInstanceUptime--Not implemented (b).
- mplsTunnelAdminStatus--Administrative status of a tunnel (c).
- mplsTunnelOperStatus--Actual operating status of a tunnel (c).
- mplsTunnelRowStatus--This object is used in conjunction with configuring a new tunnel. This object will always be seen as "active" (a).
- mplsTunnelStorageType--Storage type of a tunnel entry (c).

- mplsTunnelHopListIndexNext--Next valid index to use as an index in the mplsTunnelHopTable.
- **mplsTunnelHopTable** --Entries in this table exist only for tunnel configurations and correspond to the path options defined for the tunnel. Two types of path options exist: *explicit* and *dynamic* . This table shows all hops listed in the explicit path options, while showing only the destination hop for dynamic path options. The tunnel hop table is indexed by tunnel ID, path option, and hop number.

Following is a list and description of each table entry.

- mplsTunnelHopListIndex--Primary index into the table.
- mplsTunnelHopIndex--Secondary index into the table.
- mplsTunnelHopAddrType--Indicates if the address of this hop is the type IPv4 or IPv6.
- mplsTunnelHopIpv4Addr--The IPv4 address of this hop.
- mplsTunnelHopIpv4PrefixLen--The prefix length of the IPv4 address.
- mplsTunnelHopIpv6Addr--The IPv6 address of this hop.
- mplsTunnelHopIpv6PrefixLen--The prefix length of the IPv6 address.
- mplsTunnelHopAsNumber--This object will contain 0 or the autonomous system number of the hop, depending on the value of mplsTunnelHopAddrType.
- mplsTunnelHopLspId--This object will contain 0 or the LSPID of the tunnel, depending on the value of mplsTunnelHopAddrType.
- mplsTunnelHopType--Denotes whether this tunnel hop is routed in a strict or loose fashion.
- mplsTunnelHopRowStatus--This object is used in conjunction with the configuring of a new row in the table.
- mplsTunnelHopStorageType--The storage type of this MIB object.

- mplsTunnelResourceIndexNext--This object contains the next appropriate value to be used for mplsTunnelResourceIndex when creating entries in the mplsTunnelResourceTable
- **mplsTunnelResourceTable** --Entries in this table correspond to the "Tspec" information displayed when you execute the **show mpls traff9c-eng tunnels** command. These entries exist only for LSPs.

The tunnel resource table is indexed by address and hop number. Following the mplsTunnelResourcePointer pointer from the tunnel table is the best way to retrieve information from this table.

Following is a list and description of each table entry.

- mplsTunnelResourceIndex--The primary index into this table.
- mplsTunnelResourceMaxRate--The maximum rate, in bits per second, supported by this tunnel.
- mplsTunnelResourceMeanRate--The mean rate, in bits per second, supported by this tunnel.

- mplsTunnelResourceMaxBurstSize--The maximum burst size, in bytes, allowed by this tunnel.
- mplsTunnelResourceRowStatus--This object is used in conjunction with the configuration of a new row in the table.
- mplsTunnelResourceStorageType--The storage type of this MIB object.

- **mplsTunnelARHopTable** --Entries in this table correspond to the actual route taken by the tunnel, and whose route was successfully signaled by the network. The hops present in this table correspond to those present in the record route object (RRO) in Resource Reservation Protocol (RSVP). You can also display the information in this table by executing the **show mpls traff9c-eng tunnels** command.

The actual route hop table is indexed by address and hop number. Following the mplsTunnelARHopTableIndex pointer from the tunnel table is the best way to retrieve information from this table.

Following is a list and description of each table entry:

- - mplsTunnelARHopListIndex--The primary index into this table.
  - mplsTunnelARHopIndex--The secondary index into this table.
  - mplsTunnelARHopIpv4Addr--The IPv4 address of this hop.
  - mplsTunnelARHopIpv4PrefixLen--The prefix length of the IPv4 address.
  - mplsTunnelARHopIpv6Addr--The IPv6 address of this hop.
  - mplsTunnelARHopIpv6PrefixLen--The prefix length of the IPv6 address.
  - mplsTunnelARHopAsNumber--This object will contain 0 or the AS number of the hop, depending on the value of mplsTunnelARHopAddrType.
  - mplsTunnelARHopAddrType--The type of address for this MIB entry, either IPv4 or IPv6.
  - mplsTunnelARHopType--Denotes whether this tunnel hop is routed in a strict or loose manner.

- **mplsTunnelCHopTable** --Entries in this table correspond to the explicit route object (ERO) in RSVP, which is used to signal the LSP. The list of hops in this table will contain those hops that are computed by the constraint-based shortest path first (SPF) algorithm. In those cases where "loose" hops are specified for the tunnel, this table will contain the hops that are "filled-in" between the loose hops to complete the path. If you specify a complete explicit path, the computed hop table matches your specified path.

The computed hop table is indexed by address and hop number. Following the mplsTunnelCHopTableIndex pointer from the tunnel table is the best way to retrieve information from this table.

Following is a list and description of each table entry:

- - mplsTunnelCHopListIndex--The primary index into this table.
  - mplsTunnelCHopIndex--The secondary index into this table.
  - mplsTunnelCHopAddrType--Indicates if the address of this hop is the type IPv4 or IPv6.
  - mplsTunnelCHopIpv4Addr--The IPv4 address of this hop.
  - mplsTunnelCHopIpv4PrefixLen--The prefix length of the IPv4 address.
  - mplsTunnelCHopIpv6Addr--The IPv6 address of this hop.
  - mplsTunnelCHopIpv6PrefixLen--The prefix length of the IPv6 address.
  - mplsTunnelCHopAsNumber--This object will contain 0 or the autonomous system number of the hop, depending on the value of mplsTunnelHopAddrType.
  - mplsTunnelCHopType--Denotes whether this tunnel hop is routed in a strict or loose way.

- **mplsTunnelPerfTable** --The tunnel performance table, which augments the **mplsTunnelTable**, provides packet and byte counters for each tunnel. This table contains the following packet and byte counters:

  - mplsTunnelPerfPackets--This packet counter works only for tunnel heads.
  - mplsTunnelPerfHCPackets--This packet counter works only for tunnel heads.

- mplsTunnelPerfErrors--This packet counter works only for tunnel heads.
- mplsTunnelPerfBytes--This byte counter works for tunnel heads and tunnel midpoints, but not for tunnel tails.
- mplsTunnelPerfHCBytes--This byte counter works for tunnel heads and tunnel midpoints, but not for tunnel tails.

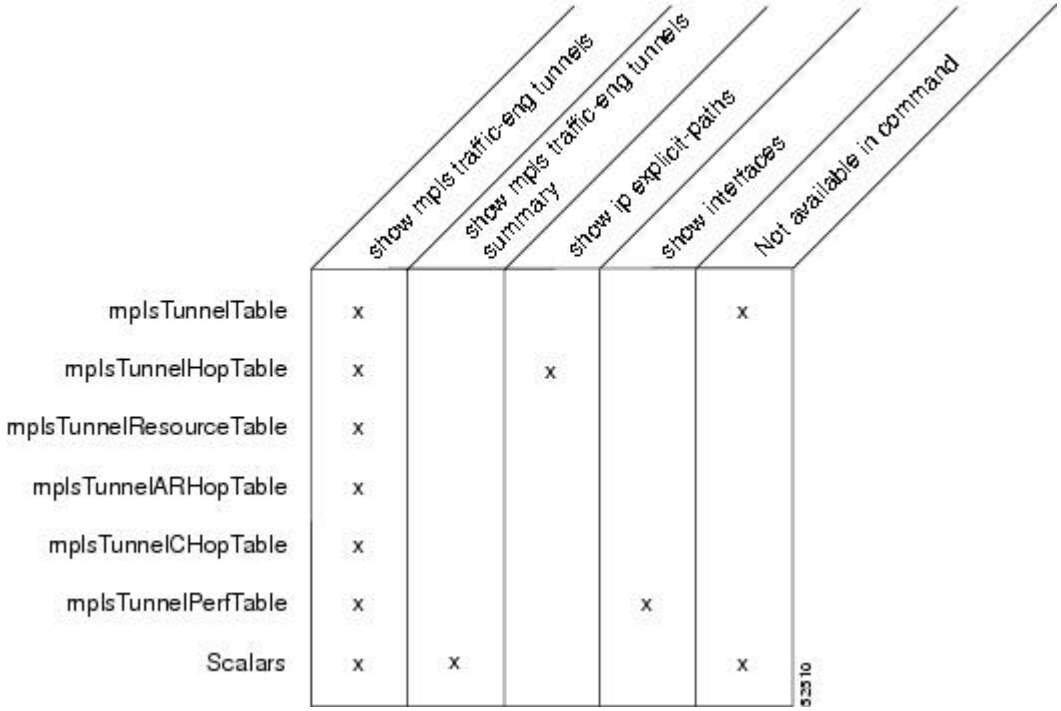
- mplsTunnelTrapEnable--The object type *mplsTunnelTrapEnable* is enhanced to be writable. Accordingly, if this object type is set to "TRUE," the following notifications are enabled, thus giving you the ability to monitor changes in the operational status of MPLS TE tunnels:
  - mplsTunnelUp
  - mplsTunnelDown
  - mplsTunnelRerouted

If the *mplsTunnelTrapEnable* object is set to "FALSE," such operational status notifications are not generated. These notification functions are based on the definitions (mplsTeNotifications) contained in the IETF draft document entitled *draft-ietf-mpls-te-mib-05.txt* .

CLI Access to MPLS Traffic Engineering MIB Information

The figure below shows commands that you can use to retrieve information from specific tables in the MPLS TE MIB. As noted in this figure, some information in the MPLS TE MIB is not retrievable by commands.

Figure 1: Commands for Retrieving MPLS TE MIB Information



Retrieving Information from the MPLS Traffic Engineering MIB

This section describes how to efficiently retrieve information about TE tunnels. Such information can be useful in large networks that contain many TE tunnels.

Traverse across a single column of the mplsTunnelTable, such as mplsTunnelName. This action provides the indexes of every tunnel configuration, and any LSPs involving the host router. Using these indexes, you can perform a GET operation to retrieve information from any column and row of the mplsTunnelTable.

The mplsTunnelTable provides pointers to other tables for each tunnel. The column mplsTunnelResourcePointer, for example, provides an object ID (OID) that you can use to access resource allocation information in the mplsTunnelResourceTable. The columns mplsTunnelHopTableIndex, mplsTunnelARHopTableIndex, and mplsTunnelCHopTableIndex provide the primary index into the mplsTunnelHopTable, mplsTunnelARHopTable, and mplsTunnelCHopTable, respectively. By traversing the MPLS TE MIB in this manner using a hop table column and primary index, you can retrieve information pertaining to the hops of that tunnel configuration.

Because tunnels are treated as interfaces, the tunnel table column (mplsTunnelIfIndex) provides an index into the Interfaces MIB that you can use to retrieve interface-specific information about a tunnel.

How to Configure the MPLS Traffic Engineering MIB

Enabling the SNMP Agent to Help Manage Various MPLS TE Tunnel Characteristics of Tunnels on the Local Router

The SNMP agent for the MPLS TE MIB is disabled by default. To enable the SNMP agent for the MPLS TE MIB, perform the following task.

SUMMARY STEPS

1. **telnet** *host*
2. **enable**
3. **show running-config**
4. **configure terminal**
5. **snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*]
6. **snmp-server enable traps** [*identification-type*] [*notification-option*]
7. **exit**
8. **write memory**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**telnet** *host* **Example:** `Router> telnet 192.172.172.172`	Telnets to the router identified by the specified IP address (represented as *xxx.xxx.xxx.xxx*).
Step 2	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Router# enable`	
Step 3	**show running-config** **Example:** `Router# show running-config`	Displays the running configuration to determine if an SNMP agent is already running. • If no SNMP information is displayed, go to Step 4 . If any SNMP information is displayed, you can modify the information or change it as needed.
Step 4	**configure terminal** **Example:** `Router# configure terminal`	Enters global configuration mode.
Step 5	**snmp-server community** *string* [**view** *view-name*] [**ro** | **rw**] [**ipv6** *nacl*] [*access-list-number*] **Example:** `Router(config)# snmp-server community comaccess ro 4`	Enables the read-only (RO) community string.
Step 6	**snmp-server enable traps** [*identification-type*] [*notification-option*] **Example:** `Router(config)# snmp-server enable traps`	Enables an LSR to send SNMP notifications or informs to an SNMP host. **Note** This command is optional. After SNMP is enabled, all MIBs (not just the TE MIB) are available for the user to query.
Step 7	**exit** **Example:** `Router(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 8	**write memory** **Example:** `Router# write memory`	Writes the modified configuration to NVRAM, permanently saving the settings.

Verifying the Status of the SNMP Agent

To verify that the SNMP agent has been enabled on a host network device, perform the following steps.

SUMMARY STEPS

1. **telnet** *host*
2. **enable**
3. **show running-config**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**telnet** *host* **Example:** `Router# telnet 192.172.172.172`	Telnets to the target device identified by the specified IP address (represented as *xxx.xxx.xxx.xxx*).
Step 2	**enable** **Example:** `Router# enable`	Enables SNMP on the target device.
Step 3	**show running-config** **Example:** `Router# show running-config`	Displays the running configuration on the target device and is used to examine the output for displayed SNMP information.

Examples

The follows example displays the running configuration on the target device and its SNMP information.

```
Router# show running-config
.
.
.
snmp-server community public ro
snmp-server community private ro
```

Any **snmp-server** statement that appears in the output and takes the form shown here verifies that SNMP has been enabled on that device.

Configuration Examples for the MPLS Traffic Engineering MIB

Example Enabling the SNMP Agent to Help Manage MPLS TE Characteristics of Tunnels on the Local Router

The following example shows how to enable an SNMP agent on a host network device:

```
Router# configure terminal
Router(config)# snmp-server community private
```

The following example shows how to enable SNMPv1 and SNMPv2C. The configuration permits any SNMP agent to access all MPLS TE MIB objects with read-only permissions using the community string public.

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

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

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

Additional References

Related Documents

Related Topic	Document Title
Cisco IOS commands	Cisco IOS Master Commands List, All Releases
Information about MPLS TE and enhancements	MPLS Traffic Engineering and Enhancements
MPLS TE commands	*Multiprotocol Label Switching Command Reference*
SNMP commands	*Network Management Command Reference*
SNMP configuration	"Configuring SNMP Support" in the *Network Management Configuration Guide*

Standards

Standard	Title
draft-ietf-mpls-te-mib-05	*MPLS Traffic Engineering Management Information Base Using SMIv2*

MIBs

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

RFCs

RFC	Title
RFC 2026	*The Internet Standards Process*

Technical Assistance

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

Feature Information for the MPLS Traffic Engineering MIB

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

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

Table 1: Feature Information for the MPLS Traffic Engineering MIB

Feature Name	Releases	Feature Information
MPLS Traffic Engineering MIB	Cisco IOS XE Release 2.3	The MPLS Traffic Engineering MIB feature enables the SNMP agent support in Cisco software for MPLS TE management, as implemented in the MPLS TE MIB. In Cisco IOS XE Release 2.3, this feature was introduced on the Cisco ASR 1000 Series Aggregation Services Routers. The following commands were introduced or modified: **snmp-server community**, **snmp-server enable traps** , **snmp-server hpst**.

Glossary

affinity bits—An MPLS traffic engineering tunnel's requirements on the attributes of the links it will cross. The tunnel's affinity bits and affinity mask must match with the attributes of the various links carrying the tunnel.

call admission precedence—An MPLS traffic engineering tunnel with a higher priority will, if necessary, preempt an MPLS traffic engineering tunnel with a lower priority. An expected use is that tunnels that are more difficult to route will have a higher priority, and can preempt tunnels that are less difficult to route, on the assumption that those lower priority tunnels can find another path.

constraint-based routing—Procedures and protocols used to determine a route across a backbone taking into account resource requirements and resource availability, instead of simply using the shortest path.

flow —A traffic load entering the backbone at one point—point of presence (POP)—and leaving it from another that must be traffic engineered across the backbone. The traffic load will be carried across one or more LSP tunnels running from the entry POP to the exit POP.

headend —The LSR at which the tunnel originates. The tunnel's "head" or tunnel interface will reside at this LSR as well.

informs —A type of notification message that is more reliable than a conventional trap notification message because an informs message requires acknowledgment.

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

label switched path (LSP) tunnel—A configured connection between two routers, using label switching to carry the packets.

LSP —label switched path. A path that is followed by a labeled packet over several hops, starting at an ingress LSR and ending at an egress LSR.

LSR —label switch router. A Layer 3 router that forwards a packet based on the value of a label encapsulated in the packet.

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

MPLS —Multiprotocol Label Switching. Switching method that forwards IP traffic using a label. This label instructs the routers and the switches in the network where to forward the packets based on preestablished IP routing information.

NMS —network management station. An NMS is 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.

notification —A message sent by an SNMP agent to a network management station, console, or terminal to indicate that a significant event within Cisco IOS software has occurred (see traps).

OSPF —Open Shortest Path First. A link-state routing protocol used for routing IP.

RSVP —Resource Reservation Protocol. Protocol for reserving network resources to provide quality of service (QoS) guarantees to application flows.

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

tailend —The downstream, receive end of a tunnel.

traffic engineering—Techniques and processes that cause routed traffic to travel through the network on a path other than the one that would have been chosen if standard routing methods were used.

trap —A message sent by an SNMP agent to a network management station, console, or terminal to indicate that a significant event within Cisco IOS software 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 notification).

VCC —virtual channel connection. A VCC is a logical circuit consisting of 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 VPI, is used to identify the next network VCL as the cell passes through a series of ATM switches on its way to its final destination.

VCL —virtual channel link. A VCL is 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 VCI, is used to identify the next network VCL as the cell passes through a series of ATM switches on its way to its final destination.