

Network Working Group
Request for Comments: 4221
Category: Informational

T. Nadeau
Cisco Systems, Inc.
C. Srinivasan
Bloomberg L.P.
A. Farrel
Old Dog Consulting
November 2005

Multiprotocol Label Switching (MPLS) Management Overview

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

A range of Management Information Base (MIB) modules has been developed to help model and manage the various aspects of Multiprotocol Label Switching (MPLS) networks. These MIB modules are defined in separate documents that focus on the specific areas of responsibility of the modules that they describe.

This document describes the management architecture for MPLS and indicates the interrelationships between the different MIB modules used for MPLS network management.

Table of Contents

1. Introduction	3
2. Terminology	3
3. The SNMP Management Framework	3
4. An Introduction to the MPLS Working Group MIB Modules	4
4.1. Structure of the MPLS MIB OID Tree	5
4.2. MPLS-TC-STD-MIB	5
4.3. MPLS-LSR-STD-MIB	5
4.4. MPLS-LDP-STD-MIB	6
4.5. MPLS-LDP-GENERIC-STD-MIB	6
4.6. MPLS-LDP-ATM-STD-MIB	6
4.7. MPLS-LDP-FRAME-RELAY-STD-MIB	7
4.8. MPLS-TE-STD-MIB	7
4.9. MPLS-FTN-STD-MIB	7

4.10.	TE-LINK-STD-MIB	7
4.11.	MIB Module Interdependencies	8
4.12.	Dependencies on External MIB Modules	9
5.	Tables, Scalars, and Notifications in MPLS-LSR-STD-MIB	10
5.1.	Tables	10
5.2.	Scalars	10
5.3.	Indexing	11
5.4.	Notifications	12
5.5.	Dependencies between MIB Module Tables	12
6.	Tables, Scalars, and Notifications in the LDP MIB	13
6.1.	MIB Modules	13
6.2.	Tables	14
6.3.	Scalars	15
6.4.	Notifications	15
6.5.	Dependencies between MIB Module Tables	15
7.	Tables, Scalars, and Notifications in MPLS-TE-STD-MIB	16
7.1.	Tables	16
7.2.	Scalars	17
7.3.	Notifications	18
7.4.	Dependencies between MIB Module Tables	18
8.	Tables, Scalars, and Notifications in MPLS-FTN-STD-MIB	18
8.1.	Tables	18
8.2.	Scalars	19
8.3.	Notifications	19
8.4.	Dependencies between MIB Module Tables	19
9.	Tables and Objects in TE-LINK-STD-MIB	19
9.1.	Tables	19
9.2.	Scalars	20
9.3.	Notifications	20
9.4.	Dependencies between MIB Module Tables	20
10.	Table Dependencies between MPLS MIB Modules	21
11.	A Note on Interfaces	21
11.1.	MPLS Tunnels as Interfaces	21
11.2.	Application of the Interfaces Group to TE Links	22
11.3.	References to Interface MIB Objects from MPLS MIB Modules	23
12.	Management Options	24
13.	Related IETF MIB Modules	25
13.1.	PWE3 Working Group MIB Modules	26
13.2.	PPVPN Working Group MIB Modules	26
13.2.1.	PPVPN-MPLS-VPN-STD-MIB	26
13.3.	CCAMP Working Group MIB Modules	26
14.	Traffic Engineering Working Group TE MIB	27
14.1.	Choosing between TE MIB Modules	27
15.	Security Considerations	28
16.	Acknowledgements	28
17.	Normative References	29
18.	Informative References	30

1. Introduction

This document describes the Management Architecture for Multi-Protocol Label Switching (MPLS) [RFC3031]. In particular, it describes how the managed objects defined in various MPLS-related Management Information Base (MIB) documents model different aspects of MPLS. Furthermore, this document explains the interactions and dependencies between each of these MIB modules.

For additional information, this document also includes a brief note on MIB modules produced by the Pseudo Wire Emulation Edge to Edge (PWE3), Provider Provisioned Virtual Private Network (PPVPN), Common Control and Measurement Plane (CCAMP), and Internet Traffic Engineering (TEWG) working groups.

The document begins with a brief outline of the SNMP framework. This is not intended to be a complete reference on SNMP, but is provided to give context to the rest of the document and to indicate reference material for readers that need to know more about SNMP.

This document does not propose any additions to the MPLS MIB framework, nor define any standards for the Internet community. It is an informational document. In all cases, the reader is advised to turn to the document that defines the MIB module in question for further information.

Comments should be made directly to the MPLS mailing list at mpls@uu.net.

2. Terminology

This document uses terminology from the MPLS architecture document [RFC3031] and the following MPLS related MIB modules: MPLS TC MIB [TCMIB], MPLS LSR MIB [LSRMIB], MPLS TE MIB [TEMIB], MPLS LDP MIB [LDPMIB], MPLS FTN MIB [FTNMIB], TE LINK MIB [TELMIB], and PPVPN MPLS VPN MIB [VPNMIB].

Throughout this document hyphenated MIB names (such as MPLS-TE-STD-MIB) should be taken to refer to specific MIB modules. Non-hyphenated MIB names (such as MPLS LDP MIB) indicate MIB documents.

3. The SNMP Management Framework

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410].

Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This document specifies a MIB module that is compliant to the SMIV2, which is described in STD 58, RFC 2578 [RFC2578], STD 58, RFC 2579 [RFC2579] and STD 58, RFC 2580 [RFC2580].

4. An Introduction to the MPLS Working Group MIB Modules

This section addresses the MIB documents produced by the MPLS working group, namely MPLS TC MIB, MPLS LSR MIB, MPLS TE MIB, MPLS LDP MIB, MPLS FTN MIB, and TE LINK MIB. The rest of this section briefly describes the following:

- the MPLS Object Identifier (OID) tree structure and the position of different MPLS related MIB modules on this tree;
- the purpose of each of the MIB modules within the MIB documents, what it can be used for, and how it relates to the other MIB modules.

Note that each MIB document contains one or more compliance statements for the modules and objects that it defines. Therefore, the support for the different MIB modules and objects is beyond the scope of this document, although some recommendations are included in the sections that follow.

4.1. Structure of the MPLS MIB OID Tree

The MPLS MIB OID tree has the following structure.

```
transmission -- RFC 2578 [RFC2578]
|
+- mplsStdMIB -- MPLS-TC-STD-MIB
|
|   +- mplsTCStdMIB -- MPLS-TC-STD-MIB
|   |
|   +- mplsLsrStdMIB -- MPLS-LSR-STD-MIB
|   |
|   +- mplsTeStdMIB -- MPLS-TE-STD-MIB
|   |
|   +- mplsLdpStdMIB -- MPLS-LDP-STD-MIB
|   |
|   +- mplsLdpAtmStdMIB -- MPLS-LDP-ATM-STD-MIB
|   |
|   +- mplsLdpFrameRelayStdMIB -- MPLS-LDP-FRAME-RELAY-STD-MIB
|   |
|   +- mplsLdpGenericStdMIB -- MPLS-LDP-GENERIC-STD-MIB
|   |
|   +- mplsFTNStdMIB -- MPLS-FTN-STD-MIB
|
+- teLinkStdMIB -- TE-LINK-STD-MIB
```

Note: The OIDs for MIB modules are assigned and managed by IANA. They can be found in the referenced MIB documents.

4.2. MPLS-TC-STD-MIB

MPLS-TC-STD-MIB defines textual conventions [RFC2579] that may be common to MPLS-related MIB modules. These conventions allow multiple MIB modules to use the same syntax and format for a concept that is shared between the MIB modules.

For example, labels are a central part of MPLS and need to be presented in many of the MIB modules. The textual convention for representing an MPLS label is defined in MPLS-TC-STD-MIB.

All of the other MPLS MIB modules import textual conventions from this MIB module.

4.3. MPLS-LSR-STD-MIB

MPLS-LSR-STD-MIB describes managed objects for modeling an MPLS Label Switching Router (LSR). This puts it at the heart of the management architecture for MPLS.

This MIB module is used to model and manage the basic label switching behavior of an MPLS LSR. It represents the label forwarding information base (LFIB) of the LSR and provides a view of the LSPs that are being switched by the LSR in question.

Since basic MPLS label switching is common to all MPLS applications, this MIB module is referenced by many of the other MPLS MIB modules.

In general, MPLS-LSR-STD-MIB provides a model of incoming labels on MPLS-enabled interfaces being mapped to outgoing labels on MPLS-enabled interfaces via a conceptual object called an MPLS cross-connect. MPLS cross-connect entries and their properties are represented in MPLS-LSR-STD-MIB and are typically referenced by other MIB modules in order to refer to the underlying MPLS LSP.

For example, MPLS-TE-STD-MIB models traffic-engineered tunnels. These tunnels map to one or more underlying MPLS LSPs. MPLS-TE-STD-MIB refers to the underlying LSPs by pointing to cross-connect entries in MPLS-LSR-STD-MIB.

4.4. MPLS-LDP-STD-MIB

MPLS-LDP-STD-MIB describes managed objects used to model and manage the MPLS Label Distribution Protocol (LDP) [RFC3036]. LDP is one of the MPLS protocols used to distribute labels and establish LSPs.

This MIB module contains objects common to all LDP implementations. For an LDP implementation that provides standard MIB support, this MIB module provides the core set of objects that are needed, along with one or more of the other LDP MIB modules from the following sections.

4.5. MPLS-LDP-GENERIC-STD-MIB

This MIB module provides objects for managing the LDP Per Platform Label Space and is typically implemented along with the MPLS-LDP-STD-MIB module. This MIB Module contains tables for configuring MPLS Generic Label Ranges. Although the LDP Specification does not provide a way to configure Label Ranges for Generic Labels, the MIB module does provide a way to reserve a range of generic labels because the working group thought this was useful.

4.6. MPLS-LDP-ATM-STD-MIB

This MIB module is typically supported along with MPLS-LDP-STD-MIB by LDP implementations if LDP uses ATM as the Layer 2 medium. Tables in this MIB module allow for configuring LDP to use ATM.

4.7. MPLS-LDP-FRAME-RELAY-STD-MIB

This MIB module is typically supported along with MPLS-LDP-STD-MIB by LDP implementations if LDP uses Frame Relay as the Layer 2 medium. Tables in this MIB module allow for configuration of LDP to use Frame Relay.

4.8. MPLS-TE-STD-MIB

MPLS-TE-STD-MIB describes managed objects that are used to model and manage MPLS Traffic Engineered (TE) Tunnels.

This MIB module is based on a table that represents TE tunnels that either originate from, traverse via, or terminate on the LSR in question. The MIB module provides configuration and statistics objects needed for TE tunnels.

4.9. MPLS-FTN-STD-MIB

MPLS-FTN-STD-MIB describes managed objects that are used to model and manage the MPLS FEC-to-NHLFE (FTN) mappings that take place at an ingress Label Edge Router (LER).

An LER is an LSR placed at the edge of an MPLS domain, and it passes traffic into and out of the MPLS domain. An ingress LER is responsible for classifying data and assigning it to a suitable LSP or tunnel.

This classification is done using Forwarding Equivalence Classes (FECs) that define the common attributes of data (usually packets) that will be treated in the same way. Once data has been classified, it can be handed off to an LSP or tunnel through the Next Hop Label Forwarding Entry (NHLFE).

In the case of an IP-to-MPLS mapping, the FEC objects describe IP 6-tuples that represent source and destination address ranges, source and destination port ranges, the IPv4 Protocol field or IPv6 next-header field, and the DiffServ Code Point (DSCP).

4.10. TE-LINK-STD-MIB

TE-LINK-STD-MIB describes managed objects that are used to model and manage TE links, including bundled links, in an MPLS network.

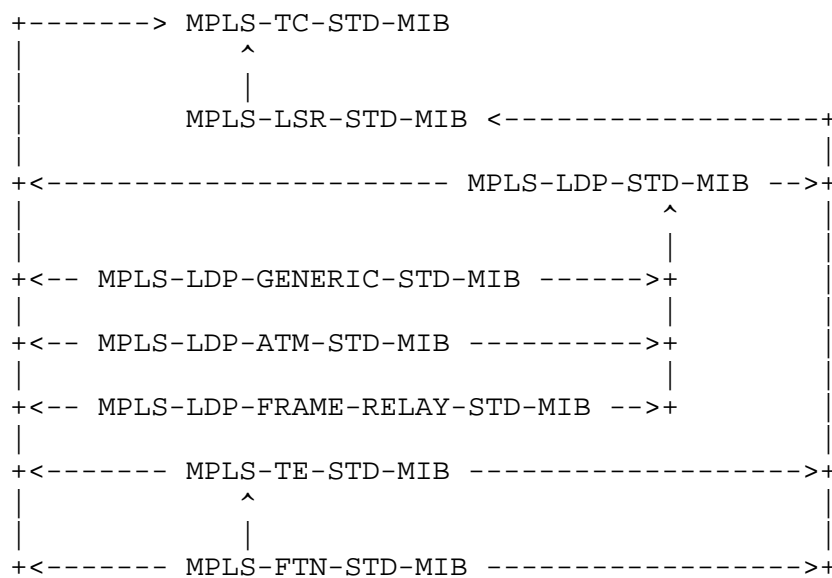
The TE link feature is designed to aggregate one or more similar data channels or TE links between a pair of LSRs. A TE link is a sub-interface capable of carrying traffic-engineered MPLS traffic.

A bundled link is a sub-interface that bonds the traffic of a group of one or more TE links.

4.11. MIB Module Interdependencies

This section provides an overview of the relationship between the MPLS MIB modules described above. More details of these relationships are given below after the MIB modules have been discussed in more detail.

The arrows in the following diagram show a 'depends on' relationship. A relationship "MIB module A depends on MIB module B" means that MIB module A uses an object, object identifier, or textual convention defined in MIB module B, or that MIB module A contains a pointer (index or RowPointer) to an object in MIB module B.



Thus:

- All the MPLS MIB modules depend on MPLS-TC-STD-MIB.
- MPLS-LDP-STD-MIB, MPLS-TE-STD-MIB, and MPLS-FTN-STD-MIB contain references to objects in MPLS-LSR-STD-MIB.
- MPLS-LDP-GENERIC-STD-MIB, MPLS-LDP-ATM-STD-MIB, and MPLS-LDP-FRAME-RELAY-STD-MIB contain references to objects in MPLS-LDP-STD-MIB.
- MPLS-FTN-STD-MIB contains references to objects in MPLS-TE-STD-MIB.

Note that there is a textual convention (MplsIndexType) defined in MPLS-LSR-STD-MIB that is imported by MPLS-LDP-STD-MIB.

4.12. Dependencies on External MIB Modules

With the exception of MPLS-TC-STD-MIB, all the MPLS MIB modules have dependencies on the Interfaces MIB [RFC2863]. MPLS-FTN-STD-MIB references IP-capable interfaces on which received traffic is to be classified using indexes in the Interface Table (ifTable) of IF-MIB [RFC2863]. The other MPLS MIB modules reference MPLS-capable interfaces in ifTable.

The Interfaces Group of IF-MIB [RFC2863] defines generic managed objects for managing interfaces. The MPLS MIB modules contain media-specific extensions to the Interfaces Group for managing MPLS interfaces.

The MPLS MIB modules assume the interpretation of the Interfaces Group to be in accordance with [RFC2863], which states that ifTable contains information on the managed resource's interfaces and that each sub-layer below the internetwork layer of a network interface is considered an interface. Thus, the MPLS interface is represented as an entry in ifTable.

The interrelation of entries in ifTable is defined by the Interfaces Stack Group defined in [RFC2863].

Additionally, MPLS-LDP-ATM-STD-MIB imports the textual convention AtmVpIdentifier from ATM-TC-MIB to represent an ATM virtual path identifier, whereas MPLS-LDP-FRAME-RELAY-STD-MIB imports the textual convention DLCI from FRAME-RELAY-DTE-MIB to represent a Data Link Channel identifier.

MPLS-LDP-STD-MIB imports the textual conventions IndexInteger and IndexIntegerNextFree from [RFC3289], and MPLS-TE-STD-MIB imports IndexIntegerNextFree. IndexInteger provides a standard arbitrary index, whereas IndexIntegerNextFree is used by a management agent that needs to select an appropriate value for an arbitrary index.

Finally, all of the MIB modules import standard textual conventions such as integers, strings, timestamps, etc., from the MIB modules in which they are defined. This is business as usual for a MIB module and is not discussed further in this document.

5. Tables, Scalars, and Notifications in MPLS-LSR-STD-MIB

5.1. Tables

MPLS-LSR-STD-MIB contains the following tables.

- The interface configuration table (`mplsInterfaceTable`) is used for enabling MPLS on MPLS-capable interfaces.
- The in-segment (`mplsInSegmentTable`) and out-segment (`mplsOutSegmentTable`) tables are used to configure and monitor LSP segments carrying data into and out of the LSR, respectively.
- The in-segment mapping table (`mplsInSegmentMapTable`) provides a look-up table that enables the discovery of an in-segment in `mplsInSegmentTable` from the known incoming interface and incoming label.
- The cross-connect table (`mplsXCTable`) is used to associate in and out segments in order to form a cross-connect (i.e., to represent an LSP transiting the LSR).
- The label stack table (`mplsLabelStackTable`) allows the specification of multi-label stacks to be imposed on a given LSP at this LSR.
- The MPLS in-segment (`mplsInSegmentPerfTable`) and out-segment (`mplsOutSegmentPerfTable`) performance tables contain objects to measure the performance of LSPs.
- The MPLS interface performance table (`mplsInterfacePerfTable`) has objects to measure MPLS performance on a per-interface basis.

5.2. Scalars

Where tables in the MIB module have arbitrary indexes, scalars are provided to supply the next available index. This applies to `mplsInSegmentTable`, `mplsOutSegmentTable`, `mplsXCTable`, and `mplsLabelStackTable`, but see the section on indexing, below.

`mplsMaxLabelStackDepth` defines the maximum size of a imposed label stack supported at this LSR (and not, as the description in MPLS-LSR-STD-MIB states, the maximum label stack depth supported by the LSR).

`mplsXCNotificationsEnable` is used to enable and disable notifications from MPLS-LSR-STD-MIB.

5.3. Indexing

Note that the indexing used by the tables in MPLS-LSR-STD-MIB is unusual. A specific textual convention, `MplsIndexType`, is defined in the MIB module and is used as the type for indexes to `mplsInSegmentTable`, `mplsOutSegmentTable`, `mplsXCTable`, and `mplsLabelStackTable`. The textual convention is defined as an octet string of between one and twenty-four octets, inclusive.

Although this convention can be used to map simple integers and so preserve the normal indexing techniques, it may also be used to encode more complex indexing rules that may be useful to implementations that subdivide their label spaces according to physical or implementation constraints (such as placing the responsibility for a subset of labels with a line card).

Note that it would be unusual, but not impossible, to make sophisticated use of these indexes in a write-access MIB since the 'next' index value would be hard to determine. Thus, non-simple values are likely only to be used in read-only MIBs in which the indexes are generated as a result of signaling protocol implementations or other configuration means. The formatting and interpretation of non-simple indexes is out of the scope of the MIB module definition and is expected to be part of the manageability statement for a particular device. When the formatting is not known by an agent, it should treat the index as a plain octet string containing an integer of between one and twenty-four octets.

As described in the previous section, scalars are provided to allow agents to discover a suitable value to use as an index when creating a new row in one of these tables. These scalars all use a second textual convention, `MplsIndexNextType`, also defined within MPLS-LSR-STD-MIB. This textual convention allows the 'null string', (that is, a string of length one octet with value 0x00). The null string is used to indicate that either write access is not supported or no more indexes are currently available.

Note that the usage of the `nextIndex` scalars is such that at any time a scalar supplies a value that is currently unused as an index to the specific table. In order to avoid lacunae in the indexing of a table under normal usage, implementations are recommended to change the value in an `nextIndex` scalar only when the index is used (that is, when a row is created) and not when the `nextIndex` scalar is read. In a 'busy' table, this may result in row creation attempts failing and agents having to re-read the scalar before making a second row creation attempt. The desire to avoid this issue is in opposition to the desire to avoid lacunae.

5.4. Notifications

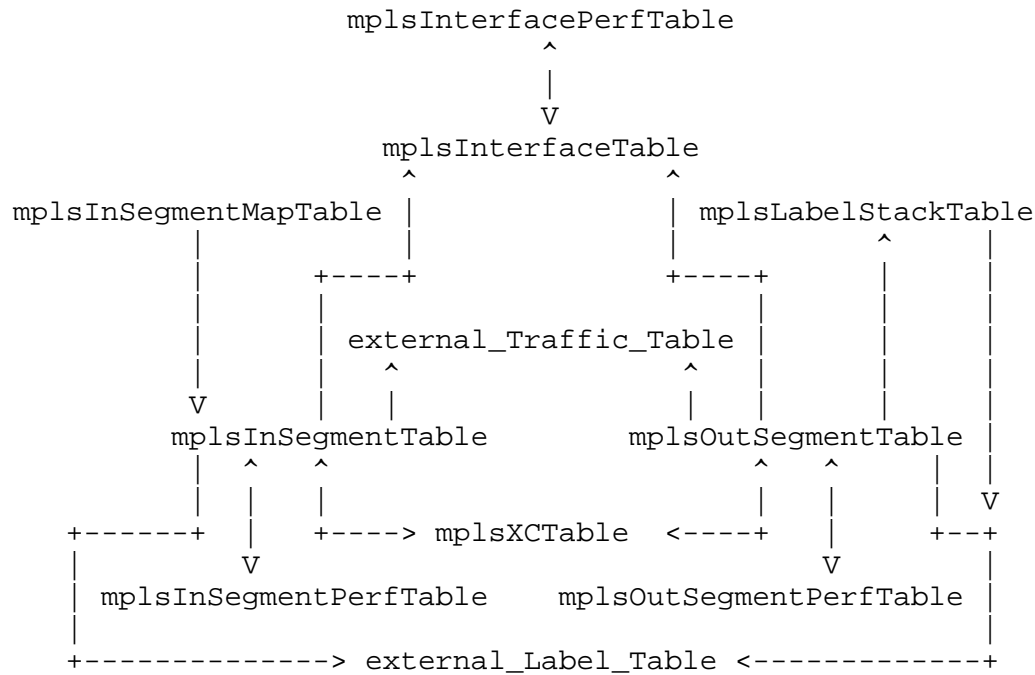
MPLS-LSR-STD-MIB can issue two notifications (if notifications are enabled).

- mplsXCUp reports when a cross-connect becomes active.
- mplsXCDown reports when a cross-connect becomes inactive.

5.5. Dependencies between MIB Module Tables

The tables in MPLS-LSR-STD-MIB are related as shown on the diagram below. The arrows indicate a reference from one table to another.

Note that the various MIB tables contain two instances of pointers to external tables that are not currently defined. Entries in an external Traffic Parameters Table (external_Traffic_Table) are pointed to using RowPointers from the mplsInSegmentTable (mplsInSegmentTrafficParamPtr) and from the mplsOutSegmentTable (mplsOutSegmentTrafficParamPtr) to allow representation of the traffic parameters for the MPLS segment. Alternatively, the pointers may indicate an entry in the Tunnel Resource Table (mplsTunnelResourceTable) in MPLS-TE-STD-MIB. Similarly, an external label table may be used to store label values if, for some reason, they are not stored in place within the LSR MIB tables. This might occur if extra per-label space information needs to be stored, and it paves the way for GMPLS where labels cannot always be stored in a 32-bit value. RowPointers are used from the mplsInSegmentTable (mplsInSegmentLabelPtr), the mplsOutSegmentTable (mplsOutSegmentTopLabelPtr), and from the mplsLabelStackTable (mplsLabelStackLabelPtr).



6. Tables, Scalars, and Notifications in the LDP MIB

6.1. MIB Modules

The MIB document for LDP contains four MIB modules. This structure makes it easier for an implementation to select only those modules that are relevant to it. The MIB Modules are MPLS-LDP-STD-MIB, MPLS-LDP-GENERIC-STD-MIB, MPLS-LDP-ATM-STD-MIB, and MPLS-LDP-FRAME-RELAY-STD-MIB.

MPLS-LDP-STD-MIB defines objects that are specific to LDP without any Layer 2 objects. MPLS-LDP-GENERIC-STD-MIB defines Layer 2 Per Platform Label Space objects for use with MPLS-LDP-STD-MIB and for use on Ethernet. MPLS-LDP-ATM-STD-MIB defines Layer 2 Asynchronous Transfer Mode (ATM) objects for use with MPLS-LDP-STD-MIB. MPLS-LDP-FRAME-RELAY-STD-MIB defines Layer 2 FRAME-RELAY objects for use with MPLS-LDP-STD-MIB.

The MPLS-LDP-STD-MIB module provides the core support and is typically supported along with at least one of the Layer 2 MIB modules.

6.2. Tables

The tables in the LDP MIB for configuring the LDP behavior of an LSR are as follows.

- The LDP Entity Table (`mplsLdpEntityTable`) provides a way to configure the LSR for using LDP. There must be at least one LDP Entity for the LSR to support LDP. Each entry/row in this table represents a single LDP Entity.
- Several tables exist to help configure LDP's use of labels. These are spread through the MIB modules described in the previous section. They are: `mplsLdpEntityGenLRTTable`, `mplsLdpEntityAtmParmsTable` and `mplsLdpEntityAtmLRTTable`, `mplsLdpEntityFrameRelayParmsTable` and `mplsLdpEntityFrLRTTable`. They are used to configure generic, ATM, and Frame Relay labels as their names suggest.
- The LDP Peer Table (`mplsLdpPeerTable`) is a read-only table that contains information about LDP Peers known to LDP Entities.
- The LDP Hello Adjacencies Table (`mplsLdpHelloAdjacencyTable`) is a table of all adjacencies between all LDP Entities and all LDP Peers.
- Several tables exist to monitor and control LDP sessions. The LDP Session Table (`mplsLdpSessionTable`) represents sessions between an LDP Entity and a Peer. `mplsLdpAtmSesTable` and `mplsLdpFrameRelaySesTable` contain session information specific to ATM.
- The MPLS LDP Session Peer Address Table (`mplsLdpSesPeerAddrTable`) stores addresses learned after session initialization via Address Message advertisement.
- The LDP FEC Table (`mplsFecTable`) represents FEC (Forwarding Equivalence Class) information that may be in use on one or more LSPs. The LDP LSP FEC Table (`mplsLdpLspFecTable`) shows the FECs associated with each LSP.
- MPLS-LDP-STD-MIB has a mapping table (`mplsLdpLspTable`) that maps the LDP MIB's representation of LDP sessions to the underlying LSR MIB's representation of the LSPs created by these sessions, by pointing to `mplsInSegmentTable`, `mplsOutSegmentTable`, and `mplsXCTable`, respectively.

- Statistics may be gathered through the LDP Entity Statistics Table (mplsLdpEntityStatsTable) and the LDP Session Statistics Table (mplsLdpSesStatsTable).

6.3. Scalars

Where tables in the MIB modules have arbitrary indexes, scalars are provided to supply the next available index. This applies to mplsLdpEntityTable and mplsFecTable.

Two scalars exist to configure the LSR. The LSR ID is set in mplsLdpLsrId, and the loop detection capabilities are reported in mplsLdpLsrLoopDetectionCapable.

6.4. Notifications

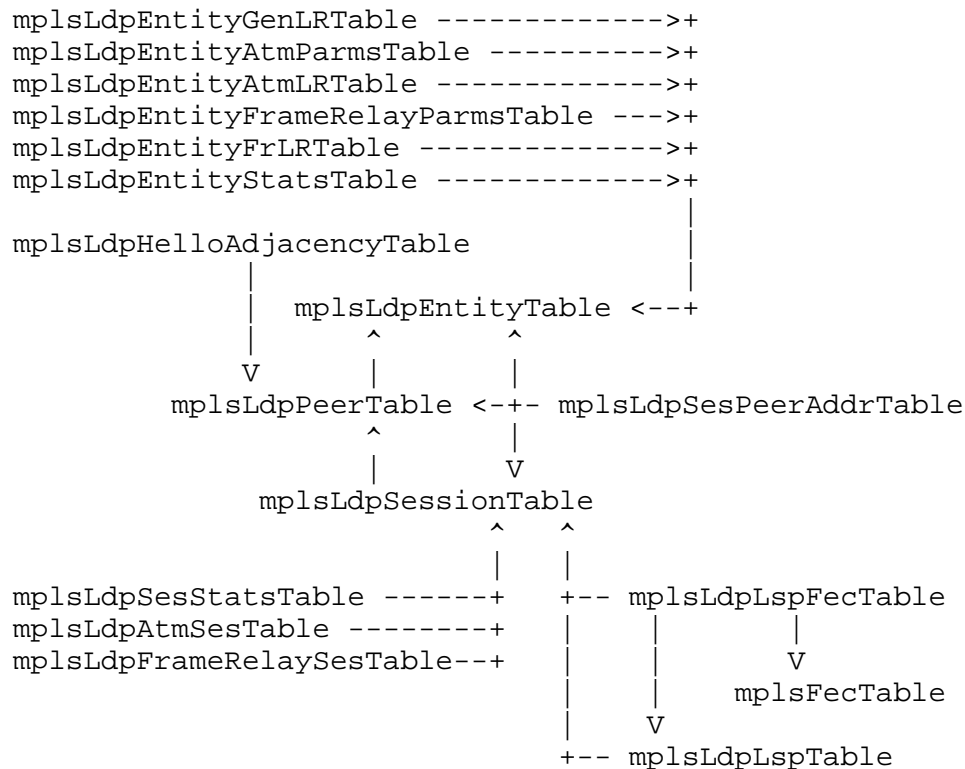
MPLS-LDP-STD-MIB defines four notifications that a device can issue.

- mplsLdpInitSesThresholdExceeded is reported when the number of Session Initialization messages exceeds a configured threshold.
- mplsLdpPVLMismatch is issued if the Path Vector Limit for a configured Entity and Peer do not match.
- mplsLdpSessionUp and mplsLdpSessionDown report the transition of Session state.

No scalar object is provided to enable and disable notifications from MPLS-LDP-STD-MIB. Instead, the implementer is referred to [RFC3413].

6.5. Dependencies between MIB Module Tables

The many tables in the four LDP MIB modules are related as shown on the diagram below. The arrows indicate a reference from one table to another. Note that in many cases the reference is through an augmentation of the referenced table.



7. Tables, Scalars, and Notifications in MPLS-TE-STD-MIB

7.1. Tables

MPLS-TE-STD-MIB contains the following tables.

- The Tunnel Table (mplsTunnelTable) is used to configure and report MPLS tunnels. Note that reporting of tunnels in this table at transit LSRs is optional.

Entries in mplsTunnelTable are indexed by four objects. The source and destination LSR IDs give context to the entry, and an index (mplsTunnelIndex) identifies the tunnel itself. However, the fourth index (mplsTunnelInstance) may give rise to some confusion since its usage is not clearly explained.

The description says: "Uniquely identifies an instance of a tunnel. It is useful to identify multiple instances of tunnels for the purposes of backup and parallel tunnels." In the case of backup tunnels, multiple instances of the same tunnel may be defined, but only one is active at any time. Different instances may have different properties (such as explicit routes), and one instance may be set up to protect against failure of another.

Parallel tunnels may be used to provide load sharing or protection.

The `mplsTunnelInstancePriority` object is used to indicate the precedence of tunnels with the same LSR IDs and `mplsTunnelIndex` value. The `mplsTunnelPrimaryInstance` object gives a quick reference back to the preferred instance of the tunnel.

The `mplsTunnelIndex` value is typically signaled as the Tunnel ID, and the `mplsTunnelInstance` as the LSP ID, in protocols where both fields exist. In protocols where there is only one identifying index (usually known as the LSP ID), only the `mplsTunnelIndex` is signaled.

- The Resource Table (`mplsTunnelResourceTable`) is used to configure resources to be requested on this tunnel. The CRLDP resource table (`mplsTunnelCRLDPResTable`) is used to request additional resource details that are specific to tunnels signaled using CR-LDP.
- The routes requested, computed, and actually used for a tunnel are found in the Tunnel Hop Table (`mplsTunnelHopTable`), Tunnel Computed Hop Table (`mplsTunnelCHopTable`), and Tunnel Actual Hop Table (`mplsTunnelARHopTable`).
- Statistics about the performance of tunnels may be gathered through the Tunnel Performance Table (`mplsTunnelPerfTable`).

7.2. Scalars

Where tables in the MIB module have arbitrary indexes, scalars are provided to supply the next available index. This applies to `mplsTunnelTable`, `mplsTunnelResourceTable`, and `mplsTunnelHopTable`.

Two scalars exist to configure the support for MPLS tunnels on the LSR. `mplsTunnelTEDistProto` lists the signaling methods and protocols supported. `mplsTunnelMaxHops` defines the size of route that may be configured on the LSR.

Two further scalars enhance the statistics on the LSR by counting the number of configured (`mplsTunnelConfigured`) and active (`mplsTunnelActive`) tunnels.

The scalar `mplsTunnelNotificationMaxRate` is used to control the rate at which notifications are issued from MPLS-TE-STD-MIB. A rate of zero means that notifications must not be issued. If notifications

would be generated faster than the configured rate, an implementation may choose to discard notifications or to queue them for distribution at a quieter time.

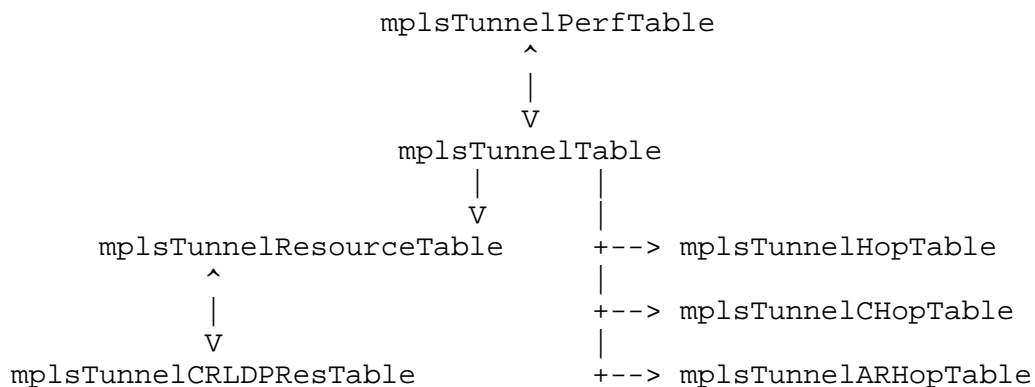
7.3. Notifications

MPLS-TE-STD-MIB defines four notifications that a device can issue. The rate of dispatch of notifications is controlled as described in the previous section.

- `mplsTunnelUp` and `mplsTunnelDown` report the transition of Tunnel state.
- Rerouting and re-optimization of Tunnels paths are reported by `mplsTunnelRerouted` and `mplsTunnelReoptimized`.

7.4. Dependencies between MIB Module Tables

The tables in MPLS-TE-STD-MIB are related as shown on the diagram below. The arrows indicate a reference from one table to another.



8. Tables, Scalars, and Notifications in MPLS-FTN-STD-MIB

8.1. Tables

MPLS-FTN-STD-MIB contains the following tables.

- The FEC-to-NHLFE Table (`mplsFTNTable`) defines the FEC to NHLFE rules to be applied to incoming packets, and the actions to be taken on matching packets.
- The FEC-to-NHLFE Mapping Table (`mplsFTNMapTable`) provides the capability to activate FTN rules defined in the `mplsFTNTable` on specific interfaces in the system.

- Performance statistics for FTN rules are found in the `mplsFTNPerfTable`.

8.2. Scalars

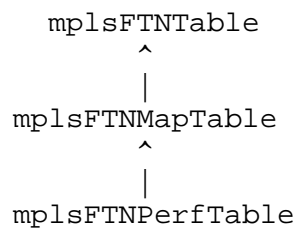
This MIB module contains the scalars `mplsFTNTableLastChanged` and `mplsFTNMapTableLastChanged` to indicate the last time an object changed in `mplsFTNTable` and `mplsFTNMapTable`, respectively. Another scalar, `mplsFTNIndexNext`, is used to supply the next valid index for creating new conceptual rows in `mplsFTNTable`.

8.3. Notifications

There are no notifications in this MIB module.

8.4. Dependencies between MIB Module Tables

The tables in MPLS-FTN-STD-MIB are related as shown on the diagram below. The arrows indicate a reference from one table to another.



9. Tables and Objects in TE-LINK-STD-MIB

9.1. Tables

TE-LINK-STD-MIB contains the following tables.

- The TE link table (`teLinkTable`) is used to specify TE links, including bundled links, and their generic traffic-engineering parameters.
- The TE link descriptor table (`teLinkDescriptorTable`) is used to list the TE link descriptors.
- The shared risk link group (SRLG) table (`teLinkSrlgTable`) is used to specify the SRLGs associated with TE links.
- The TE link bandwidth table (`teLinkBandwidthTable`) is used to report priority-based bandwidth values associated with TE links.

- The component link table (componentLinkTable) is used to identify the data-bearing component links that are associated with the TE links and specify the data-bearing link generic traffic engineering parameters.
- The component link descriptor table (componentLinkDescriptorTable) is used to list the data-bearing component link descriptors.
- The component link bandwidth table (componentLinkBandwidthTable) is used to report priority-based bandwidth values associated with data-bearing component links.

9.2. Scalars

There are no scalars in this MIB module.

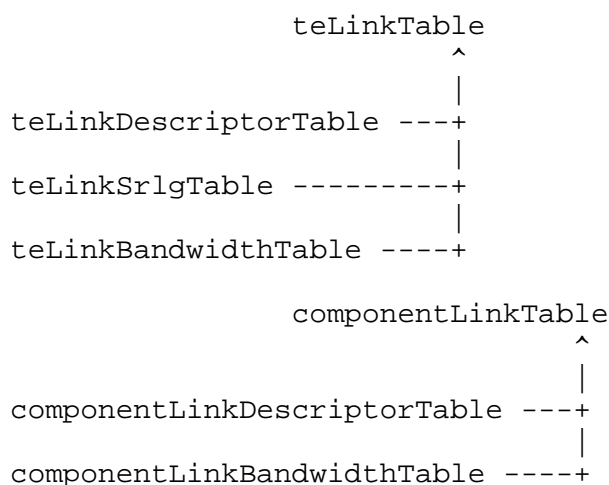
9.3. Notifications

There are no notifications in this MIB module.

9.4. Dependencies between MIB Module Tables

The tables in TE-LINK-STD-MIB are related as shown on the diagram below. The arrows indicate a reference from one table to another.

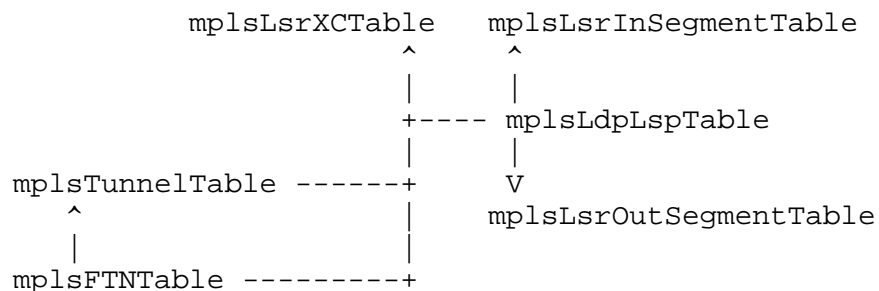
Note that many of the associations between tables are through a common index that is the ifIndex of the related interface.



10. Table Dependencies between MPLS MIB Modules

Section 4.11 gave an overview of how the MPLS MIB modules are related. Now that the tables in the MIB modules have been introduced, it is possible to give a more detailed diagram of these relationships.

MPLS-TC-STD-MIB is left off the diagram because many of the MIB module tables use textual conventions from that MIB module.



11. A Note on Interfaces

The Interfaces Group of IF-MIB [RFC2863] defines generic managed objects for managing interfaces. The MPLS MIB modules make references to interfaces so that it can be clearly determined where the procedures managed by the MIB modules should be performed. Additionally, the MPLS MIB modules (notably MPLS-TE-STD-MIB and TE-LINK-STD-MIB) utilize interface stacking within the Interface Group.

11.1. MPLS Tunnels as Interfaces

MPLS-TE-STD-MIB builds on the concept of managing MPLS Tunnels as logical interfaces. [RFC2863] states that the interfaces table (ifTable) contains information on the managed resource's interfaces, and that each sub-layer below the internetwork layer of a network interface is considered an interface. Thus, an MPLS Tunnel managed as an interface is represented as an entry in the ifTable. The interrelation of entries in the ifTable is defined by the Interfaces Stack Group defined in [RFC2863].

When using MPLS Tunnels as interfaces, the interface stack table might appear as follows:

```

+-----+
| MPLS tunnel interface ifType = mplsTunnel(150) |
+-----+
|           MPLS interface ifType = mpls(166)           |
+-----+
|                   Underlying layer                   |
+-----+

```

In the diagram above, "Underlying layer" refers to the ifIndex of any interface type for which MPLS internetworking has been defined. Examples include ATM, Frame Relay, and Ethernet.

A detailed listing of the mapping between ifTable objects and their use for MPLS Tunnels is given in [TEMIB]. A few key objects are listed here to provide an overview of the concepts.

Each MPLS tunnel is represented by an entry in the ifTable. Each tunnel is therefore assigned a unique ifIndex.

The type of an interface represented by an entry in the ifTable is indicated by the ifType object. The value that is allocated to identify an MPLS tunnel is 150.

The ifOperStatus object reflects the actual operational status of the MPLS tunnel and may be mapped from the mplsTunnelOperStatus object.

It may be considered convenient and good management to set the ifName object to reflect the name of the MPLS tunnel as contained in the mplsTunnelName object.

11.2. Application of the Interfaces Group to TE Links

TE-LINK-STD-MIB also uses interface stacking to manage TE Link interfaces as logical interfaces. The TE Link interface is represented as an entry in the ifTable. The interrelation of entries in the ifTable is defined by Interfaces Stack Group defined in [RFC2863]. When using TE Link interfaces, the interface stack table might appear as follows:

+-----+-----+	
MPLS interface ifType = mpls(166)	
ifIndex = 1	
+-----+-----+	
TE link (bundled link) ifType = teLink(200)	
ifIndex = 2	
+-----+-----+	
TE link ifType = teLink(200)	TE link ifType = teLink(200)
ifIndex = 3	ifIndex = 4
+-----+-----+	
Component link	Component link
ifType = opticalTransport(196)	ifType = opticalTransport(196)
ifIndex = 5	ifIndex = 6
+-----+-----+	

In the above diagram, "opticalTransport" is an example of an underlying physical interface: in this case an optical transport interface. TE link management and bundling can be seen in the levels of interface stacking. Two TE links are defined, each managing an optical transport link. These two TE links are combined into a bundle, which is managed as a single TE link interface. This TE Link interface supports MPLS and is presented as an MPLS interface.

A detailed listing of the mapping between ifTable objects and their use for TE Links is given in [TELMIB]. A few key objects are listed here to provide an overview of the concepts.

Each TE Link interface is represented by a separate entry in the ifTable, with a unique ifIndex.

The type of an interface represented by an entry in the ifTable is indicated by the ifType object. The value that is allocated to identify a TE Link is 200.

11.3. References to Interface MIB Objects from MPLS MIB Modules

MPLS-TE-STD-MIB contains two objects that reference the management of an MPLS tunnel as an interface. mplsTunnelIsIf is a TruthValue that indicates whether the tunnel is present in the ifTable. If the tunnel is managed as an interface, the mplsTunnelIfIndex object contains the ifIndex that identifies the corresponding entry in the ifTable.

MPLS-LSR-STD-MIB includes a table (mplsInterfaceTable) for configuring the support for MPLS on specific interfaces. A conceptual row in this table is created automatically by an LSR for every interface that is capable of and configured for support of MPLS. A conceptual row in this table will exist if and only if a

corresponding entry in ifTable exists with ifType = mpls(166). The fate of the entries in the two tables are closely linked so that if the entry in the ifTable is operationally disabled, the entry in mplsInterfaceTable is deleted. During the life of an entry in mplsInterfaceTable, a corresponding entry is managed in mplsInterfacePerfTable to show performance counters for the MPLS-capable interface.

The ifIndex that identifies MPLS-capable interfaces also plays an important indexing role in MPLS-LSR-STD-MIB. In-segments (that is, incoming LSP labels) are represented in mplsInSegmentTable, which is indexed by the mplsInSegmentIfIndex and mplsInSegmentLabel objects. mplsInSegmentIfIndex is set to the ifIndex of the incoming MPLS-capable interface. mplsInSegmentLabel identifies the incoming MPLS label. Note that the corresponding mplsOutSegmentTable contains an mplsOutSegmentIfIndex object to identify the outgoing MPLS-capable interface, but that this does not form part of the index of the table.

MPLS-LDP-STD-MIB uses ifIndex extensively to identify the interface over which MPLS is active.

Within MPLS-FTN-STD-MIB, mplsFTNMapTable maps entries in mplsFTNTable to interfaces on which mplsFTNTable entries should be activated. Interfaces are identified using their ifIndex values.

12. Management Options

It is not the intention of this document to provide instructions or advice to implementers of Management Stations, Management Agents, or managed entities. It is, however, useful to make some observations about how the MIB modules described above might be used to manage MPLS systems.

All MPLS LSPs may appear in MPLS-LSR-STD-MIB. At transit nodes, they are seen as full cross-connects between incoming labels on incoming interfaces and outgoing labels on outgoing interfaces. At ingress or egress points, the cross-connections are unbalanced having spoof upstream or downstream legs, respectively.

Split and merge points of LSPs may be represented as more complex cross-connects in MPLS-LSR-STD-MIB. Similarly, bidirectional LSPs can be represented by using the same cross-connect index for each of the forward and reverse cross-connections.

The modules in the LDP MIB are intended solely for use with LDP and CR-LDP. LSPs that are signaled through other means may conveniently be stored in mplsLdpLspTable for consistency with LSPs set up using

LDP, but there is little further value to this because the table gives only pointers into MPLS-LSR-STD-MIB. If, however, the LSPs are established with associated FECs using some signaling method other than LDP (for example, BGP), it may be advantageous to use `mplsLdpLspTable`, `mplsFecTable`, and `mplsLdpLspFecTable` to correlate the LSPs.

Note that if CR-LDP is the signaling protocol, there is no requirement to use the LSP-related tables in the LDP MIB since the LSP will be adequately represented in MPLS-TE-MIB and MPLS-LSR-STD-MIB.

MPLS tunnels may be represented in MPLS-TE-STD-MIB with their cross-connects indicated in MPLS-LSR-STD-MIB. Tunnels are often (although not always) set up with a series of constraints that may be represented in MPLS-TE-STD-MIB. Note that a distinguishing feature of a tunnel is that it has an ingress and an egress, where LSPs established through LDP may be end-to-end or may be hop-by-hop.

All LSPs (tunnels and non-tunnels) may be established as a result of signaling protocols already defined or for future study. In addition, LSPs may be set up manually by issuing configuration commands to each of the LSRs on the LSP. These commands may utilize SNMP by performing SET operations to the MIB module tables and objects described here. Alternatively, configuration may be through some non-standard interface such as a Command Line or a Graphical User Interface. Such configured LSPs may also be represented in the MIB module tables.

Do not be misled by considerations of the "permanence" of LSPs when deciding which tables of which MIB modules to use. An MPLS tunnel may have a very long life expectancy if it is set up by an amnesiac user. Otherwise, it may have a very short lifetime if it is automatically provisioned to satisfy on-demand traffic requirements. Similarly, an LSP established in response to a routing protocol (sometimes known as a hop-by-hop LSP) may be equally stable or unstable.

13. Related IETF MIB Modules

This section describes the broad interactions between MIB modules produced by the PWE3, PPVPN, and CCAMP working groups and the MPLS MIB modules. This information is provided as background and is not central to this document.

13.1. PWE3 Working Group MIB Modules

The PWE3 working group has produced a document [PWE3FW] that includes a description of the framework for MIB modules within PWE3 operation. Since the PWE3 architecture includes the use of MPLS as an emulated service and as a PSN service, the MPLS MIB modules described above may be leveraged. The PWE3 framework document describes the interactions between the MPLS MIB modules and the PWE3 MIB modules.

13.2. PPVPN Working Group MIB Modules

At present, the PPVPN working group has not included a discussion of how the MPLS MIB modules interact with the MIB modules being produced by that working group. The authors of this document hope to make a forthcoming addition to the PPVPN framework document [PPVPNFW] detailing these interactions. At the moment, there are two MIB modules, [VPNMIB] and [VPNTCMIB], which are discussed next.

13.2.1. PPVPN-MPLS-VPN-STD-MIB

PPVPN-MPLS-VPN-STD-MIB describes managed objects that are used to model and manage RFC2547bis MPLS VPNs [RFC2547Bis]. This MIB module contains tables that model virtual routing forwarding entries (VRFs), as well as the interfaces associated with those VRFs.

13.2.1.1. Position in the OID Tree

```
transmission -- RFC 2578 [RFC2578]
|
+- vpnMIB -- PPVPN-MPLS-VPN-STD-MIB
```

13.2.1.2. Dependencies

This MIB module currently has no direct dependencies on any of the MPLS MIB modules. This MIB module models MPLS VPN interfaces as entries in the Interfaces MIB's Interfaces Table (ifTable). This MIB module may be modified in the future to import textual conventions from MPLS-TC-STD-MIB.

A specific textual conventions MIB module [VPNTCMIB] defines textual conventions that are imported into PPVPN-MPLS-VPN-STD-MIB.

13.3. CCAMP Working Group MIB Modules

The CCAMP working group is developing MIB modules in support of GMPLS that interact directly with the MPLS MIB modules. Along with any MIB modules produced by the CCAMP working group, a separate CCAMP-

specific Management Framework document is expected to be issued describing the relationship between these MIB modules and the existing MPLS (and other) MIB modules.

14. Traffic Engineering Working Group TE MIB

The TEWG has produced a traffic engineering MIB (TE-MIB) [TEWGMIB] containing objects for monitoring traffic-engineered tunnels at their ingress points.

In many senses TE-MIB contains the same information as MPLS-TE-STD-MIB. Both MIB modules can be used to monitor MPLS tunnels; however, TE-MIB is minimalistic and caters best to TE tunnels as tunnels, at the expense of not having many advanced features of MPLS-TE-STD-MIB, whereas MPLS-TE-STD-MIB can deconstruct tunnels into hop-by-hop cross-connects, at the expense of more complexity.

The TE-MIB module imports textual conventions from the MPLS-TC-STD-MIB module and therefore is dependent on that document.

14.1. Choosing between TE MIB Modules

TE-MIB is a flexible MIB module designed to manage traffic engineering tunnels regardless of the implementation technology. This flexibility and a focus on simplicity lead to some compromises.

- Some MPLS configuration parameters are left out. For example, the resource management in TE-MIB is confined to bandwidth, so missing the full IntServ control.
- Other TE-MIB parameters are present but with only limited options; for example, the ability to configure different label distribution methods per LSP.

Extensibility of TE-MIB to related concepts (such as DiffServ and Fast Reroute) and integrations with other MIB modules (such as that in MPLS-LSR-STD-MIB) are not work items at the time of writing. The MPLS MIB modules are more closely integrated as described in this document.

Write/create access to TE-MIB is only available at the ingress, where it can be used to configure an ingress to signal a tunnel with constraints. It cannot be used to configure hop-by-hop cross-connects to build a tunnel.

The purpose of TE-MIB module is to allow a Management Agent to configure tunnels, and to inspect and monitor all tunnels (however created) at their ingress points. It does not provide information

about tunnels at any other point in the network (that is, at transit or egress nodes). This module can be used, for example, to configure the constraints of a tunnel, whereupon the ingress would compute the tunnel path and signal it. The MIB module can then be used at the ingress to monitor the tunnel's path(s), their status, and the tunnel's uptime and counters. This MIB module is not designed to configure hop-by-hop cross-connects to build a tunnel.

15. Security Considerations

This document describes the interrelationships amongst the different MIB modules relevant to MPLS management and as such does not have any security implications in and of itself.

Each specific MIB document specifies specific MIB objects, and such a document must provide a proper security considerations section that explains the security aspects of those objects.

The attention of readers is particularly drawn to the security implications of making MIB objects available for create or write access through an access protocol such as SNMP. SNMPv1 by itself is an insecure environment. Even if the network itself is made secure (for example, by using IPsec), there is no control over who on the secure network is allowed to access and GET (read) the objects in this MIB. It is recommended that the implementers consider the security features as provided by the SNMPv3 framework. Specifically, the use of the User-based Security Model STD 62, RFC 3414 [RFC3414], and the View-based Access Control Model STD 62, RFC 3415 [RFC3415], is recommended.

It is then a customer/user responsibility to ensure that the SNMP entity giving access to an instance of this MIB is properly configured to give access to only those objects, and to those principals (users) that have legitimate rights to access them.

16. Acknowledgements

Many small pieces of text in this document have been borrowed from the documents that define the MIB modules described here. The authors would like to express appreciation to all who worked on those MIB documents.

Thanks also to all those who attended the November 2002 MPLS MIB open meeting and gave constructive feedback, and in particular to Sharon Chisholm for her thoughts on Management Options.

Thanks to Kireeti Kompella for revising the text on TE-MIB.

Without the consistent pressure and encouragement from Bert Wijnen, this document would not have been written.

17. Normative References

- [FTNMIB] Nadeau, T., Srinivasan, C., and A. Viswanathan, "Multiprotocol Label Switching (MPLS) Forwarding Equivalence Class To Next Hop Label Forwarding Entry (FEC-To-NHLFE) Management Information Base (MIB)", RFC 3814, June 2004.
- [LDPMIB] Cucchiara, J., Sjostrand, H., and J. Luciani, "Definitions of Managed Objects for the Multiprotocol Label Switching (MPLS), Label Distribution Protocol (LDP)", RFC 3815, June 2004.
- [LSRMIB] Srinivasan, C., Viswanathan, A., and T. Nadeau, "Multiprotocol Label Switching (MPLS) Label Switching Router (LSR) Management Information Base (MIB)", RFC 3813, June 2004.
- [RFC2863] McCloghrie, K. and F. Kastenholtz, "The Interfaces Group MIB ", RFC 2863, June 2000.
- [RFC3289] Baker, F., Chan, K., and A. Smith, "Management Information Base for the Differentiated Services Architecture", RFC 3289, May 2002.
- [TCMIB] Nadeau, T. and J. Cucchiara, "Definitions of Textual Conventions (TCs) for Multiprotocol Label Switching (MPLS) Management", RFC 3811, June 2004.
- [TELMIB] Dubuc, M., Dharanikota, S., Nadeau, T., J. Lang, "Traffic Engineering Link Management Information Base", RFC 4220, November 2005.
- [TEMIB] Srinivasan, C., Viswanathan, A., and T. Nadeau, "Multiprotocol Label Switching (MPLS) Traffic Engineering (TE) Management Information Base (MIB)", RFC 3812, June 2004.

18. Informative References

- [PPVPNFW] Callon, R. and M. Suzuki, "A Framework for Layer 3 Provider-Provisioned Virtual Private Networks (PPVPNs)", RFC 4110, July 2005.
- [PWE3FW] Bryant, S. and P. Pate, "Pseudo Wire Emulation Edge-to-Edge (PWE3) Architecture", RFC 3985, March 2005.
- [RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.
- [RFC2547Bis] Rosen, E., et al., "MPLS/BGP VPNs", Work in Progress, October 2002.
- [RFC2578] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.
- [RFC3036] Andersson, L., Doolan, P., Feldman, N., Fredette, A., and B. Thomas, "LDP Specification", RFC 3036, January 2001.
- [RFC3410] Case, J., Mundy, R., Partain, D. and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3413] Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications", STD 62, RFC 3413, December 2002.
- [RFC3414] Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)", STD 62, RFC 3414, December 2002.

- [RFC3415] Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)", STD 62, RFC 3415, December 2002.
- [TEWGMIB] Kompella, K., "A Traffic Engineering (TE) MIB", RFC 3970, January 2005.
- [VPNMIB] Nadeau, T., et al., "MPLS/BGP Virtual Private Network Management Information Base Using SMIV2", Work in Progress, November 2002.
- [VPNTCMIB] Schliesser, B. and T. Nadeau, "Definition of Textual Conventions for Provider Provisioned Virtual Private Network (PPVPN) Management", Work in Progress, November 2002.

Authors' Addresses

Thomas D. Nadeau
Cisco Systems, Inc.
1414 Massachusetts Ave.
Boxborough, MA 01719

EMail: tnadeau@cisco.com

Cheenu Srinivasan
Bloomberg L.P.
731 Lexington Avenue
New York, NY 10022

Phone: (212) 617-3682
EMail: cheenu@bloomberg.net

Adrian Farrel
Old Dog Consulting

Phone: +44 (0) 1978 860944
EMail: adrian@olddog.co.uk

Full Copyright Statement

Copyright (C) The Internet Society (2005).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

