This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects to configure and/or monitor a Generalized Multiprotocol Label Switching (GMPLS) Label Switching Router (LSR).
1. Introduction

This memo defines a portion of the Management Information Base (MIB) for use with network management protocols in the Internet community. In particular, it describes managed objects for modeling a Generalized Multiprotocol Label Switching (GMPLS) [GMPLSARCH] Label Switching Router (LSR).

Comments should be made directly to the CCAMP mailing list at ccamp@ops.ietf.org.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119, reference [RFC2119].

1.1. Migration Strategy

There are two MIB modules in this document. The GMPLS LSR MIB module extends the LSR MIB module defined for use with MPLS [LSRMIB]. The only changes made are additions for support of GMPLS or changes that are necessary to support the increased complexity of a GMPLS system. The GMPLS Label MIB module may be referenced using a row pointer from objects within the LSR MIB module.
The companion document modeling and managing GMPLS based traffic engineering [GMPLSTEMIB] extends the MPLS TE MIB module [TEMIB] with the same intentions.

Textual conventions and OBJECT-IDENTIFIERS are defined in [GMPLSTCMIB] which extends the set of textual conventions originally defined in [TCMIB].
2. Terminology

This document uses terminology from the document describing the MPLS architecture [RFC3031] and the GMPLS architecture [GMPLSArch].

A label switched path (LSP) is modeled as a connection consisting of one or more incoming segments (in-segments) and/or one or more outgoing segments (out-segments) at an LSR. The association or interconnection of the in-segments and out-segments is accomplished by using a cross-connect. We use the terminology "connection" and "LSP" interchangeably where the meaning is clear from the context.

- **in-segment**: This is analogous to a GMPLS label on an interface.
- **out-segment**: This is analogous to a GMPLS label on an interface.
- **cross-connect**: This describes the conceptual connection between a set of in-segments and out-segments. Note that either set may be empty; for example, a cross-connect may connect only out-segments together with no in-segments in the case where an LSP is originating on an LSR.

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 memo 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. Outline

Configuring statically provisioned GMPLS LSPs through an LSR involves the following steps:

- Configuring an interface using the MPLS LSR MIB module.
- Enabling GMPLS on GMPLS capable interfaces using this MIB module.
- Configuring in-segments and out-segments using the MPLS LSR MIB
module.

- Configuring GMPLS extensions to the in-segments and out-segments using this MIB module.

- Setting up the cross-connect table in the MPLS LSR MIB module to associate segments and/or to indicate connection origination and termination.
- Optionally setting up labels in the label table in this MIB module if the textual convention MplsLabel is not capable of holding the required label (for example, if the label requires more than 32 bits to encode it), or if the operator wishes to disambiguate GMPLS label types.

- Optionally specifying label stack actions in the MPLS LSR MIB module.

- Optionally specifying segment traffic parameters in the MPLS LSR MIB module.

4.1 MIB Modules

There are two MIB modules defined in this document.

The GMPLS LSR MIB module contains tables that extend tables defined in the MPLS LSR MIB module. This MIB module is used in conjunction with the MPLS LSR MIB module in systems that support GMPLS.

The GMPLS Label MIB module contains objects for managing GMPLS labels when they cannot be represented using the textual conventions of the MPLS TC MIB module, or when more detailed access to the sub-fields of the labels is required.

4.1.1 Summary of the GMPLS LSR MIB Module

The MIB tables in this MIB module are as follows.

- The interface configuration table (gmplsInterfaceTable), which extends mplsInterfaceTable to enable the GMPLS protocol on MPLS-capable interfaces.

- The in-segment (gmplsInSegmentTable) and out-segment (gmplsOutSegmentTable) tables extend mplsInSegmentTable and mplsOutSegmentTable to configuring GMPLS-specific parameters for LSP segments at an LSR.

These tables are described in the subsequent sections.

4.1.2 Summary of the GMPLS Label MIB Module

There is one MIB table in this MIB module as follows.
The gmplsLabelTable allows Generalized Labels to be defined and managed in a central location. Generalized Labels can be of variable length and have distinct bit-by-bit interpretations according to the use that is made of them.

These tables are described in the subsequent sections.
5. Bidirectional LSPs

This MIB module supports bidirectional LSPs as required for GMPLS. A single value of mplsXCIndex is shared by all of the segments for the entire bidirectional LSP. This facilitates a simple reference from [TEMIB] and [GMPLSTEMIB], and makes fate-sharing more obvious.

It is, however, important that the direction of segments is understood to avoid connecting all in-segments to all out-segments. This is achieved by an object in each segment that indicates the direction of the segment with respect to data flow.

A segment that is marked as 'forward' carries data from the 'head' of the LSP to the 'tail'. A segment marked as 'reverse' carries data in the reverse direction.

Where an LSP is signaled using a conventional signaling protocol, the 'head' of the LSP is the source of the signaling (also known as the ingress) and the 'tail' is the destination (also known as the egress). For manually configured LSPs an arbitrary decision must be made about which segments are 'forward' and which 'reverse'. For consistency this decision should be made across all LSRs that participate in the LSP by assigning 'head' and 'tail' ends to the LSP.

6. Example of LSP Setup

In this section we provide a brief example of using the MIB objects described in sections 7 and 8 to set up an LSP. While this example is not meant to illustrate every nuance of the MIB, it is intended as an aid to understanding some of the key concepts. It is meant to be read after going through the MIB itself. A prerequisite is an understanding of [LSRMIB].

Suppose that one would like to manually create a best-effort, bi-directional LSP. Assume that, in the forward direction, the LSP enters the LSR via MPLS interface A with ifIndex 12 and exits the LSR via MPLS interface B with ifIndex 13. For the reverse direction, we assume the LSP enters via interface B and leaves via interface A (i.e. the forward and reverse directions use the same bi-directional interfaces). Let us also assume that we do not wish to have a label stack beneath the top label on the outgoing labeled packets. The following example illustrates which rows and corresponding objects might be created to accomplish this.

We must first create rows in the gmplsLabelTable corresponding to the labels required for each of the forward and reverse direction in- and out-segments. For the purpose of this example the forward and reverse labels on each interface will be the same, hence we need to
create just two rows in the gmplsLabelTable - one for each interface.
In gmplsLabelTable:
{
  gmplsLabelInterface = 12,
gmplsLabelIndex = 1,
gmplsLabelSubindex = 0,
gmplsLabelType = gmplsFreeformGeneralizedLabel(3),
gmplsLabelFreeform = 0x123456789ABCDEF0
  gmplsLabelRowStatus = createAndGo(4)
}

In gmplsLabelTable:
{
  gmplsLabelInterface = 13,
gmplsLabelIndex = 1,
gmplsLabelSubindex = 0,
gmplsLabelType = gmplsFreeformGeneralizedLabel(3),
gmplsLabelFreeform = 0xFEDCBA9876543210
  gmplsLabelRowStatus = createAndGo(4)
}

We must next create the appropriate in-segment and out-segment entries. These are done in \cite{LSRMIB} using the mplsInSegmentTable and mplsOutSegmentTable. Note that we use a row pointer to the two rows in the gmplsLableTable rather than specifying the labels explicitly in the in- and out-segment tables. Also note that the row status for each row is set to createAndWait(5) to allow corresponding entries in the gmplsInSegmentTable and gmplsOutSegmentTable to be created.

For the forward direction.

In mplsInSegmentTable:
{
  mplsInSegmentIndex = 0x00000015
  mplsInSegmentLabel = 0, -- incoming label in label table
  mplsInSegmentNPop = 1,
  mplsInSegmentInterface = 12, -- incoming interface

  -- RowPointer MUST point to the first accessible column.
  mplsInSegmentTrafficParamPtr = 0.0,

  mplsInSegmentLabelPtr = gmplsLabelTable (12, 1, 0)
  mplsInSegmentRowStatus = createAndWait(5)
}

In mplsOutSegmentTable:
{
  mplsOutSegmentIndex = 0x00000012,
  mplsOutSegmentInterface = 13, -- outgoing interface
  mplsOutSegmentPushTopLabel = true(1),
  mplsOutSegmentTopLabel = 0, -- outgoing label in label table
-- RowPointer MUST point to the first accessible column.
mplsOutSegmentTrafficParamPtr = 0.0,
mplsOutSegmentLabelPtr = gmplsLabelTable (13, 1, 0)
mplsOutSegmentRowStatus = createAndWait(5)
}
For the reverse direction.

In mplsInSegmentTable:
\{ 
  mplsInSegmentIndex = 0x00000016

  mplsInSegmentLabel = 0, -- incoming label in label table
  mplsInSegmentNPop = 1,
  mplsInSegmentInterface = 13, -- incoming interface

  -- RowPointer MUST point to the first accesible column.
  mplsInSegmentTrafficParamPtr = 0.0,
  mplsInSegmentLabelPtr = gmplsLabelTable (13, 1, 0)

  mplsInSegmentRowStatus = createAndWait(5)
\}

In mplsOutSegmentTable:
\{ 
  mplsOutSegmentIndex = 0x00000013,
  mplsOutSegmentInterface = 12, -- outgoing interface
  mplsOutSegmentPushTopLabel = true(1),
  mplsOutSegmentTopLabel = 0, -- outgoing label in label table

  -- RowPointer MUST point to the first accesible column.
  mplsOutSegmentTrafficParamPtr = 0.0,
  mplsOutSegmentLabelPtr = gmplsLabelTable (12, 1, 0)

  mplsOutSegmentRowStatus = createAndWait(5)
\}

These table entries are extended by entries in gmplsInSegmentTable and gmplsOutSegmentTable. Note that the nature of the ‘extends’ relationship is that the entry in gmplsInSegmentTable has the same index values as the entry in mplsInSegmentTable. Similarly, the entry in gmplsOutSegmentTable has the same index values as the entry in mplsOutSegmentTable.

First for the forward direction:

In gmplsInSegmentTable(0x00000015)
\{ 
  gmplsInSegmentDirection = forward (1)
\}
In gmplsOutSegmentTable(0x00000012)
\{
gmplsOutSegmentDirection = forward (1)

Next for the reverse direction:

In gmplsInSegmentTable(0x00000016)
{
    gmplsInSegmentDirection = reverse (2)
}

Nadeau, Srinivasan, Farrel, Hall and Harrison
In `gmplsOutSegmentTable(0x00000013)`
{
    gmplsOutSegmentDirection = reverse (2)
}

Next, two cross-connect entries are created in the `mplsXCTable` of the MPLS LSR MIB, thereby associating the newly created segments together.

In `mplsXCTable`:
{
    mplsXCIndex = 0x01,
    mplsXCInSegmentIndex = 0x00000015,
    mplsXCOutSegmentIndex = 0x00000012,

    mplsXCLspId = 0x0102 -- unique ID
    mplsXCLabelStackIndex = 0x00, -- only a single outgoing label
    mplsXCRowStatus = createAndGo(4)
}

In `mplsXCTable`:
{
    mplsXCIndex = 0x02,
    mplsXCInSegmentIndex = 0x00000016,
    mplsXCOutSegmentIndex = 0x00000013,

    mplsXCLspId = 0x0102 -- unique ID
    mplsXCLabelStackIndex = 0x00, -- only a single outgoing label
    mplsXCRowStatus = createAndGo(4)
}

Finally, the in-segments and out-segments are activated.

In `mplsInSegmentTable(0x00000015)`:
{
    mplsInSegmentRowStatus = active(1)
}

In `mplsInSegmentTable(0x00000016)`:
{
    mplsInSegmentRowStatus = active(1)
}

In `mplsOutSegmentTable(0x00000012)`:
{
    mplsOutSegmentRowStatus = active(1)
}
In mplsOutSegmentTable(0x00000013):
{
    mplsOutSegmentRowStatus = active(1)
}
7. GMPLS Label Switching Router MIB Definitions

GMPLS-LSR-STD-MIB DEFINITIONS ::= BEGIN

IMPORTS
   MODULE-IDENTITY, OBJECT-TYPE, Unsigned32, zeroDotZero
   FROM SNMPv2-SMI
   MODULE-COMPLIANCE, OBJECT-GROUP
   FROM SNMPv2-CONF
   RowPointer
   FROM SNMPv2-TC
   GmplsSegmentDirection, gmplsStdMIB
   FROM GMPLS-TC-STD-MIB
   mplsInterfaceIndex, mplsInSegmentIndex, mplsOutSegmentIndex
   FROM MPLS-LSR-STD-MIB
;

gmplsLsrStdMIB MODULE-IDENTITY
   LAST-UPDATED
   "200402130900Z" -- 13 February 2004 9:00:00 GMT
   ORGANIZATION
   "Common Control And Management Protocols (CCAMP) Working Group"
   CONTACT-INFO
   "
   Thomas D. Nadeau
   Cisco Systems, Inc.
   Email: tnadeau@cisco.com

   Cheenu Srinivasan
   Bloomberg L.P.
   Email: cheenu@bloomberg.net

   Adrian Farrel
   Old Dog Consulting
   Email: adrian@olddog.co.uk

   Ed Harrison
   Data Connection Ltd.
   Email: ed.harrison@dataconnection.com

   Tim Hall
   Data Connection Ltd.
   Email: tim.hall@dataconnection.com

   Comments about this document should be emailed direct to the
   CCAMP working group mailing list at ccamp@ops.ietf.org"

DESCRIPTION
   "This MIB module contains managed object definitions for the Generalized
   Multiprotocol Label Switching...
(GMPLS) Router as defined in: Mannie et al.,
Generalized Multi-Protocol Label Switching (GMPLS)
Architecture, draft-ietf-ccamp-gmpls-
arquitectura-07.txt, May 2003, work in progress.

Copyright (C) The Internet Society (2004). This
version of this MIB module is part of RFCXXX; see
the RFC itself for full legal notices."
-- Revision history.
REVISION
"200402130900Z" -- 13 February 2004 09:00:00 GMT

DESCRIPTION
"Initial revision, published as part of RFC XXXX."
::= { gmplsStdMIB xx }

-- Top level components of this MIB module.

-- Notifications
-- no notifications are currently defined.
gmplsLsrNotifications OBJECT IDENTIFIER ::= { gmplsLsrStdMIB 0 }

-- Tables, Scalars
gmplsLsrObjects OBJECT IDENTIFIER ::= { gmplsLsrStdMIB 1 }

-- Conformance
gmplsLsrConformance OBJECT IDENTIFIER ::= { gmplsLsrStdMIB 2 }

-- GMPLS Interface Table.

gmplsInterfaceTable OBJECT-TYPE
SYNTAX        SEQUENCE OF GmplsInterfaceEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
"This table specifies per-interface GMPLS capability and associated information. It extends the information in mplsInterfaceTable."
::= { gmplsLsrObjects 1 }

gmplsInterfaceEntry OBJECT-TYPE
SYNTAX        GmplsInterfaceEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
"A conceptual row in this table is created automatically by an LSR for every interface capable of supporting GMPLS and which is configured to do so. A conceptual row in this table will exist if and only if a corresponding entry in mplsInterfaceTable exists, and a corresponding entry in ifTable exists with ifType = mpls(166). If the associated entry in ifTable is operationally disabled (thus removing the GMPLS capabilities on the interface) or the entry in mplsInterfaceTable is deleted, the corresponding entry in this table MUST be deleted shortly"
thereafter.

The indexing is the same as that for mplsInterfaceTable. Thus, the entry with index 0 represents the per-platform label space and contains parameters that apply to all interfaces that participate in the per-platform label space."

INDEX { mplsInterfaceIndex }
::= { gmplsInterfaceTable 1 }
GmplsInterfaceEntry ::= SEQUENCE {
    gmplsInterfaceSignalingCaps      BITS,
    gmplsInterfaceRsvpHelloPeriod    Unsigned32
}

gmplsInterfaceSignalingCaps OBJECT-TYPE
SYNTAX      BITS {
    unknown (0),
    rsvpGmpls (1),
    crldpGmpls (2), -- note the use of CR-LDP is deprecated
    otherGmpls (3)
}
MAX-ACCESS  read-create
STATUS       current
DESCRIPTION   "Defines the signaling capabilities on this interface. Multiple bits may legitimately be set at once. Setting no bits implies that GMPLS signaling cannot be performed on this interface and all LSPs must be manually provisioned."
::= { gmplsInterfaceEntry 1 }

gmplsInterfaceRsvpHelloPeriod OBJECT-TYPE
SYNTAX      Unsigned32
UNITS        "milliseconds"
MAX-ACCESS  read-create
STATUS       current
DESCRIPTION   "Period, in milliseconds, between sending RSVP Hello messages on this interface. A value of 0 indicates that no Hello messages should be sent on this interface."
DEFVAL       { 3000 }
::= { gmplsInterfaceEntry 2 }

-- End of gmplsInterfaceTable

-- In-segment table.

gmplsInSegmentTable  OBJECT-TYPE
SYNTAX        SEQUENCE OF GmplsInSegmentEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION   "This table extends the mplsInSegmentTable to provide GMPLS-specific information about incoming segments to an LSR."
::= { gmplsLsrObjects 2 }

gmplsInSegmentEntry  OBJECT-TYPE
SYNTAX        GmplsInSegmentEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION

"An entry in this table extends the representation of an incoming segment represented by an entry in mplsInSegmentTable. An entry can be created by a network administrator or an SNMP agent, or a GMPLS signaling protocol."
Note that the storage type for this entry SHOULD be inherited from the corresponding entry in the mplsInSegmentTable given by the value of the mplsInSegmentStorageType object.

INDEX { mplsInSegmentIndex }
::= { gmplsInSegmentTable 1 }

GmplsInSegmentEntry ::= SEQUENCE {
  gmplsInSegmentDirection        GmplsSegmentDirection,
  gmplsInSegmentExtraParamsPtr   RowPointer
}

gmplsInSegmentDirection OBJECT-TYPE
SYNTAX        GmplsSegmentDirection
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
  "This object indicates the direction of data flow on this segment. This object cannot be modified if mplsInSegmentRowStatus for the associated entry in the mplsInSegmentTable is active(1)."
DEFVAL        { forward }
::= { gmplsInSegmentEntry 1 }

gmplsInSegmentExtraParamsPtr  OBJECT-TYPE
SYNTAX       RowPointer
MAX-ACCESS   read-create
STATUS       current
DESCRIPTION
  "Some Tunnels will run over transports that can usefully support technology-specific additional parameters (for example, SONET resource usage). Such can be supplied from an external table and referenced from here. A value of zeroDotZero in this attribute indicates that there is no such additional information."
DEFVAL      { zeroDotZero }
::= { gmplsInSegmentEntry 2 }

-- End of gmplsInSegmentTable

-- Out-segment table.

gmplsOutSegmentTable  OBJECT-TYPE
SYNTAX        SEQUENCE OF GmplsOutSegmentEntry
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
  "This table extends the mplsOutSegmentTable to provide GMPLS-specific information about outgoing segments from an LSR."
::= { gmplsLsrObjects 3 }
gmplsOutSegmentEntry OBJECT-TYPE
SYNTAX       GmplsOutSegmentEntry
MAX-ACCESS   not-accessible
STATUS       current
DESCRIPTION  "An entry in this table extends the representation of an outgoing segment represented by an entry in mplsOutSegmentTable. An entry can be created by a network administrator or an SNMP agent, or a GMPLS signaling protocol. Note that the storage type for this entry SHOULD be inherited from the corresponding entry in the mplsOutSegmentTable given by the value of the mplsOutSegmentStorageType object."
INDEX  { mplsOutSegmentIndex }
 ::= { gmplsOutSegmentTable 1 }

GmplsOutSegmentEntry ::= SEQUENCE {
  gmplsOutSegmentDirection       GmplsSegmentDirection,
  gmplsOutSegmentTTLDecrement    Unsigned32,
  gmplsOutSegmentExtraParamsPtr  RowPointer
}

gmplsOutSegmentDirection OBJECT-TYPE
SYNTAX       GmplsSegmentDirection
MAX-ACCESS   read-create
STATUS       current
DESCRIPTION  "This object indicates the direction of data flow on this segment. This object cannot be modified if mplsOutSegmentRowStatus for the associated entry in the mplsOutSegmentTable is active(1)."
DEFVAL  { forward }
 ::= { gmplsOutSegmentEntry 1 }

gmplsOutSegmentTTLDecrement OBJECT-TYPE
SYNTAX       Unsigned32 (0..255)
MAX-ACCESS   read-create
STATUS       current
DESCRIPTION  "This object indicates the amount by which to decrement the TTL of any payload packets forwarded on this segment if per-hop decrementing is being done.

A value of zero indicates that no decrement should be made or that per-hop decrementing is not in force. See the gmplsTunnelTTLDecrement object in the gmplsTunnelTable of [GMPLSTEMIB] for a value by which to decrement the TTL for the whole of a tunnel.
This object cannot be modified if
mplsOutSegmentRowStatus for the associated entry in
the mplsOutSegmentTable is active(1)."
DEFVAL   { 0 }
::= { gmplsOutSegmentEntry 2 }
gmplsOutSegmentExtraParamsPtr  OBJECT-TYPE
SYNTAX       RowPointer
MAX-ACCESS   read-create
STATUS       current
DESCRIPTION
 "Some Tunnels will run over transports that can
 usefully support technology-specific additional parameters
 (for example, SONET resource usage). Such can be supplied from
 an external table and referenced from here.
 A value of zeroDotZero in this attribute indicates that there
 is no such additional information."
DEFVAL { zeroDotZero }
 ::= { gmplsOutSegmentEntry 3 }

-- End of gmplsOutSegmentTable

-- Module compliance.

gmplsLsrGroups
 OBJECT IDENTIFIER ::= { gmplsLsrConformance 1 }

gmplsLsrCompliances
 OBJECT IDENTIFIER ::= { gmplsLsrConformance 2 }

-- Compliance requirement for fully compliant implementations.

gmplsLsrModuleFullCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
 "Compliance statement for agents that provide full
 support for GMPLS-LSR-STD-MIB."

MODULE IF-MIB -- The Interfaces Group MIB, RFC 2863.

MANDATORY-GROUPS {
   ifGeneralInformationGroup,
   ifCounterDiscontinuityGroup
}

MODULE MPLS-LSR-STD-MIB -- The MPLS LSR MIB

MANDATORY-GROUPS {
   mplsInterfaceGroup,
   mplsInSegmentGroup,
   mplsOutSegmentGroup,
   mplsXCGroup,
   mplsPerfGroup,
mplsLsrNotificationGroup
}

MODULE -- this module

MANDATORY-GROUPS  {
  gmplsInterfaceGroup,
  gmplsInSegmentGroup,
  gmplsOutSegmentGroup
}

Nadeau, Srinivasan, Farrel, Hall and Harrison
-- gmplsInSegmentTable

OBJECT      gmplsInSegmentDirection
SYNTAX      GmplsSegmentDirection
MIN-ACCESS  read-write
DESCRIPTION
"Only forward(1) needs to be supported by
implementations that only support unidirectional
LSPs."

-- gmplsOutSegmentTable

OBJECT      gmplsOutSegmentDirection
SYNTAX      GmplsSegmentDirection
MIN-ACCESS  read-write
DESCRIPTION
"Only forward(1) needs to be supported by
implementations that only support unidirectional
LSPs."

OBJECT      gmplsOutSegmentTTLDecrement
SYNTAX      Unsigned32 (0..255)
MIN-ACCESS  read-only
DESCRIPTION
"Write access is not required."

::= { gmplsLsrCompliances 1 }

-- Compliance requirement for implementations that provide read-only
-- access.

gmplsLsrModuleReadOnlyCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"Compliance requirement for implementations that only
provide read-only support for GMPLS-LSR-STD-MIB. Such
devices can then be monitored but cannot be configured
using this MIB modules."

MODULE IF-MIB -- The interfaces Group MIB, RFC 2863

MANDATORY-GROUPS {
ifGeneralInformationGroup,
ifCounterDiscontinuityGroup
}

MODULE MPLS-LSR-STD-MIB
MANDATORY-GROUPS {
    mplsInterfaceGroup,
    mplsInSegmentGroup,
    mplsOutSegmentGroup,
    mplsXCGroup,
    mplsPerfGroup
}
MODULE -- this module

MANDATORY-GROUPS {
    gmplsInterfaceGroup,
    gmplsInSegmentGroup,
    gmplsOutSegmentGroup
}

-- gmplsInterfaceGroup

OBJECT gmplsInterfaceSignalingCaps
SYNTAX BITS {
    unknown (0),
    rsvpGmpls (1),
    crldpGmpls (2),
    otherGmpls (3)
} MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT gmplsInterfaceRsvpHelloPeriod
SYNTAX Unsigned32 MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

-- gmplsInSegmentTable

OBJECT gmplsInSegmentDirection
SYNTAX GmplsSegmentDirection MIN-ACCESS read-only
DESCRIPTION
"Write access is not required. Only forward(1) needs
   to be supported by implementations that only support
   unidirectional LSPs."

OBJECT gmplsInSegmentExtraParamsPtr
SYNTAX RowPointer MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

-- gmplsOutSegmentTable

OBJECT gmplsOutSegmentDirection
SYNTAX GmplsSegmentDirection MIN-ACCESS read-only
DESCRIPTION

"Write access is not required. Only forward(1) needs to be supported by implementations that only support unidirectional LSPs."

Nadeau, Srinivasan, Farrel, Hall and Harrison

[Page 16]
OBJECT gmplsOutSegmentTTLDecrement
SYNTAX  Unsigned32 (0..255)
MIN-ACCESS read-only
DESCRIPTION
   "Write access is not required."

OBJECT gmplsOutSegmentExtraParamsPtr
SYNTAX  RowPointer
MIN-ACCESS read-only
DESCRIPTION
   "Write access is not required."

::= { gmplsLsrCompliances 2 }

-- Units of conformance.

gmplsInterfaceGroup OBJECT-GROUP
OBJECTS {
   gmplsInterfaceSignalingCaps,
   gmplsInterfaceRsvpHelloPeriod
}
STATUS current
DESCRIPTION
   "Collection of objects needed for GMPLS interface
    configuration and performance information."
::= { gmplsLsrGroups 1 }

gmplsInSegmentGroup OBJECT-GROUP
OBJECTS {
   gmplsInSegmentDirection,
   gmplsInSegmentExtraParamsPtr
}
STATUS current
DESCRIPTION
   "Collection of objects needed to implement a GMPLS
    in-segment."
::= { gmplsLsrGroups 2 }

gmplsOutSegmentGroup OBJECT-GROUP
OBJECTS {
   gmplsOutSegmentDirection,
   gmplsOutSegmentTTLDecrement,
   gmplsOutSegmentExtraParamsPtr
}
STATUS current
DESCRIPTION
   "Collection of objects needed to implement a GMPLS
    out-segment."
::= { gmplsLsrGroups 3 }
8. GMPLS Label MIB Definitions

GMPLS-LABEL-STD-MIB DEFINITIONS ::= BEGIN

IMPORTS
   MODULE-IDENTITY, OBJECT-TYPE, Unsigned32,
   Integer32
         FROM SNMPv2-SMI
   MODULE-COMPLIANCE, OBJECT-GROUP
         FROM SNMPv2-CONF
   RowStatus, StorageType
         FROM SNMPv2-TC
   InterfaceIndexOrZero
         FROM IF-MIB
   IndexIntegerNextFree
         FROM DIFFSERV-MIB
   MplsLabel
         FROM MPLS-TC-STD-MIB
   GmplsGeneralizedLabelTypes, GmplsFreeformLabel,
   gmplsStdMIB
         FROM GMPLS-TC-STD-MIB
;

gmplsLabelStdMIB MODULE-IDENTITY
     LAST-UPDATED
         "200402130900Z" -- 13February 2004 9:00:00 GMT
     ORGANIZATION
         "Common Control And Management Protocols (CCAMP)
          Working Group"
     CONTACT-INFO
         "
           Thomas D. Nadeau
           Cisco Systems, Inc.
           Email: tnadeau@cisco.com

           Cheenu Srinivasan
           Bloomberg L.P.
           Email: cheenu@bloomberg.net

           Adrian Farrel
           Old Dog Consulting
           Email: adrian@olddog.co.uk

           Ed Harrison
           Data Connection Ltd.
           Email: ed.harrison@dataconnection.com

           Tim Hall
           Data Connection Ltd.
           Email: tim.hall@dataconnection.com
Comments about this document should be emailed direct to the CCAMP working group mailing list at ccamp@ops.ietf.org

DESCRIPTION

"This MIB module contains managed object definitions for labels within GMPLS systems."
-- Revision history.

REVISION
"200402130900Z" -- 13 February 2004 09:00:00 GMT

DESCRIPTION
"Initial revision, published as part of RFC XXXX."

::= { gmplsStdMIB xx }

-- Top level components of this MIB module.

-- Notifications
-- no notifications are currently defined.

gmplsLabelNotifications OBJECT IDENTIFIER ::= { gmplsLabelStdMIB 0 }

-- Tables, Scalars

gmplsLabelObjects OBJECT IDENTIFIER ::= { gmplsLabelStdMIB 1 }

-- Conformance

gmplsLabelConformance OBJECT IDENTIFIER ::= { gmplsLabelStdMIB 2 }

-- GMPLS Label Table.

gmplsLabelIndexNext OBJECT-TYPE
SYNTAX IndexIntegerNextFree
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object contains an unused value for
gmplsLabelIndex, or a zero to indicate
that no unused value exists or is available.

An management application wishing to create
a row in the gmplsLabelTable may read this
object and then attempt to create a row in
the table. If row creation fails (because
another application has already created a row
with the supplied index) the management
application should read this object again
to get a new index value.

When a row is created in the gmplsLabelTable
with the gmplsLabelIndex value held by this
object, an implementation MUST change the value
in this object.
"

::= { gmplsLabelObjects 1 }

gmplsLabelTable OBJECT-TYPE
SYNTAX SEQUENCE OF GmplsLabelEntry
"Table of GMPLS Labels. This table allows the representation of the more complex label forms required for GMPLS which cannot be held within the textual convention MplsLabel. That is labels that cannot be encoded within 32 bits. It is, nevertheless also capable of holding 32 bit labels or regular MPLS labels if desired.
Each entry in this table represents an individual GMPLS label value. Labels in the tables in other MIBs are referred to using row pointer into this table. The indexing of this table provides for arbitrary indexing and also for concatenation of labels.

::= { gmplsLabelObjects 2 }

gmplsLabelEntry OBJECT-TYPE
SYNTAX GmplsLabelEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION "An entry in this table represents a single label value. There are three indexes into the table.
- The interface index may be helpful to distinguish which labels are in use on which interfaces or to handle cases where there are a very large number of labels in use in the system. When label representation is desired to apply to the whole system or when it is not important to distinguish labels by their interfaces, this index MAY be set to zero.
- The label index provides a way of identifying the label.
- The label sub-index is only used for concatenated labels. It identifies each component label. When non-concatenated labels are used, this index SHOULD be set to zero.

A storage type object is supplied to control the storage type for each entry, but implementations should note that the storage type of conceptual rows in other tables that include row pointers to an entry in this table SHOULD dictate the storage type of the rows in this table where the row in the other table is more persistent."

INDEX {
    gmplsLabelInterface,
    gmplsLabelIndex,
    gmplsLabelSubindex }
::= { gmplsLabelTable 1 }

GmplsLabelEntry ::= SEQUENCE {
    gmplsLabelInterface          InterfaceIndexOrZero,
    gmplsLabelIndex              Unsigned32,
    gmplsLabelSubindex           Unsigned32,
    gmplsLabelType               GmplsGeneralizedLabelTypes,
    gmplsLabelMplsLabel          MplsLabel,
    gmplsLabelPortWavelength     Unsigned32,
    gmplsLabelFreeform           GmplsFreeformLabel,
    gmplsLabelSonetSdhSignalIndex Integer32,
    gmplsLabelSdhVc              Integer32,
    gmplsLabelSdhVcBranch       Integer32,
gmplsLabelSonetSdhBranch  Integer32,
gmplsLabelSonetSdhGroupBranch  Integer32,
gmplsLabelWavebandId  Unsigned32,
gmplsLabelWavebandStart       Unsigned32,
gmplsLabelWavebandEnd         Unsigned32,
gmplsLabelRowStatus           RowStatus,
gmplsLabelStorageType         StorageType
}

gmplsLabelInterface OBJECT-TYPE
SYNTAX        InterfaceIndexOrZero
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
"The interface on which this label is used. If the
label has or could have applicability across the
whole system, this object SHOULD be set to zero."
::= { gmplsLabelEntry 1 }

gmplsLabelIndex OBJECT-TYPE
SYNTAX        Unsigned32 (0..4294967295)
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
"An arbitrary index into the table to identify a label.
Note that implementations that are representing 32 bit
labels within this table MAY choose to align this index
with the value of the label, but should be aware of the
implications of sparsely populated tables.
A management application may read the gmplsLabelIndexNext
object to find a suitable value for this object."
::= { gmplsLabelEntry 2 }

gmplsLabelSubindex OBJECT-TYPE
SYNTAX        Unsigned32 (0..4294967295)
MAX-ACCESS    not-accessible
STATUS        current
DESCRIPTION
"In conjunction with gmplsLabelInterface and gmplsLabelIndex,
this object uniquely identifies this row. This sub-index
allows a single GMPLS label to be defined as a concatenation
of labels. This is particularly useful in TDM.
The ordering of sub-labels is strict with the sub-label with
lowest gmplsLabelSubindex appearing first. Note that all sub-
labels of a single GMPLS label must share the same
gmplsLabelInterface and gmplsLabelIndex values. For labels
that are not composed of concatenated sub-labels, this value
SHOULD be set to zero."
::= { gmplsLabelEntry 3 }

gmplsLabelType OBJECT-TYPE
SYNTAX        GmplsGeneralizedLabelTypes
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
"Identifies the type of this label. Note that this object
does not determine whether MPLS or GMPLS signaling is in use: a value of gmplsMplsLabel (1) denotes that a 23 bit MPLS packet label is present, but does not describe whether this is signaled using MPLS or GMPLS.
The value of this object helps determine which of the following objects are valid. This object cannot be modified if gmplsLabelRowStatus is active(1).

```plaintext
::= { gmplsLabelEntry 4 }
```

**gmplsLabelMplsLabel**

**SYNTAX**
MplsLabel

**MAX-ACCESS**
read-create

**STATUS**
current

**DESCRIPTION**
"The value of an MPLS label (that is a packet label) if this table is used to store it. This may be used in MPLS systems even though the label values can be adequately stored in the MPLS MIB modules. Further, in mixed MPLS and GMPLS systems it may be advantageous to store all labels in a single label table. Lastly, in GMPLS systems where packet labels are used (that is in systems that use GMPLS signaling and GMPLS labels for packet switching) it may be desirable to use this table.

This object is only valid if gmplsLabelType is set to gmplsMplsLabel (1).

This object cannot be modified if gmplsLabelRowStatus is active(1)."

**DEFVAL**
{ 0 }

```plaintext
::= { gmplsLabelEntry 6 }
```

**gmplsLabelPortWavelength**

**SYNTAX**
Unsigned32

**MAX-ACCESS**
read-create

**STATUS**
current

**DESCRIPTION**
"The value of a Port or Wavelength Label when carried as a Generalized Label. Only valid if gmplsLabelType is set to gmplsPortWavelengthLabel(2).

This object cannot be modified if gmplsLabelRowStatus is active(1)."

**DEFVAL**
{ 0 }

```plaintext
::= { gmplsLabelEntry 7 }
```

**gmplsLabelFreeform**

**SYNTAX**
GmplsFreeformLabel

**MAX-ACCESS**
read-create

**STATUS**
current

**DESCRIPTION**
"The value of a freeform Generalized Label that does not conform to one of the standardized label encoding or that an implementation chooses to represent as an octet string without further decoding. Only valid if gmplsLabelType is set to gmplsFreeformGeneralizedLabel(3).

This object cannot be modified if
gmplsLabelRowStatus is active(1).
 ::= { gmplsLabelEntry 8 }

Nadeau, Srinivasan, Farrel, Hall and Harrison [Page 22]
gmplsLabelSonetSdhSignalIndex OBJECT-TYPE
SYNTAX        Integer32 (0..4095)
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
"The Signal Index value (S) of a SONET or SDH
Generalized Label. Zero indicates that this field is
not significant. Only valid if gmplsLabelType is set
to gmplsSonetLabel(4) or gmplsSdhLabel(5).
This object cannot be modified if
gmplsLabelRowStatus is active(1)."
DEFVAL        { 0 }
::= { gmplsLabelEntry 9 }

gmplsLabelSdhVc OBJECT-TYPE
SYNTAX        Integer32 (0..15)
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
"The VC Indicator (U) of an SDH Generalized Label.
Zero indicates that this field is non-significant. Only valid if gmplsLabelType is set to
gmplsSdhLabel(5).
This object cannot be modified if
gmplsLabelRowStatus is active(1)."
DEFVAL        { 0 }
::= { gmplsLabelEntry 10 }

gmplsLabelSdhVcBranch OBJECT-TYPE
SYNTAX        Integer32 (0..15)
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
"The VC Branch Indicator (K) of an SDH Generalized Label.
Zero indicates that this field is non-significant. Only valid if gmplsLabelType is set to
gmplsSdhLabel(5).
This object cannot be modified if
gmplsLabelRowStatus is active(1)."
DEFVAL        { 0 }
::= { gmplsLabelEntry 11 }

gmplsLabelSonetSdhBranch OBJECT-TYPE
SYNTAX        Integer32 (0..15)
MAX-ACCESS    read-create
STATUS        current
DESCRIPTION
"The Branch Indicator (L) of a SONET or SDH
Generalized Label. Zero indicates that this field is
non-significant. Only valid gmplsLabelType is set to
gmplsSonetLabel(4) or gmplsSdhLabel(5).
This object cannot be modified if
gmplsLabelRowStatus is active(1)."
DEFVAL        { 0 }
::= { gmplsLabelEntry 12 }
gmplsLabelSonetSdhGroupBranch OBJECT-TYPE
SYNTAX Integer32 (0..15)
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The Group Branch Indicator (M) of a SONET or SDH Generalized Label. Zero indicates that this field is non-significant. Only valid if gmplsLabelType is set to gmplsSonetLabel(4) or gmplsSdhLabel(5). This object cannot be modified if gmplsLabelRowStatus is active(1)."
DEFVAL { 0 }
::= { gmplsLabelEntry 13 }

gmplsLabelWavebandId OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The waveband identifier component of a waveband label. Only valid if gmplsLabelType is set to gmplsWavebandLabel(6). This object cannot be modified if gmplsLabelRowStatus is active(1)."
DEFVAL { 0 }
::= { gmplsLabelEntry 14 }

gmplsLabelWavebandStart OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The starting label component of a waveband label. Only valid if gmplsLabelType is set to gmplsWavebandLabel(6). This object cannot be modified if gmplsLabelRowStatus is active(1)."
DEFVAL { 0 }
::= { gmplsLabelEntry 15 }

gmplsLabelWavebandEnd OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION "The end label component of a waveband label. Only valid if gmplsLabelType is set to gmplsWavebandLabel(6). This object cannot be modified if gmplsLabelRowStatus is active(1)."
DEFVAL { 0 }
::= { gmplsLabelEntry 16 }
gmplsLabelRowStatus OBJECT-TYPE
SYNTAX RowStatus
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This variable is used to create, modify, and/or delete a row in this table. When a row in this table has a row in the active(1) state, no objects in this row can be modified except the gmplsLabelRowStatus and gmplsLabelStorageType."

::= { gmplsLabelEntry 17 }

gmplsLabelStorageType OBJECT-TYPE
SYNTAX StorageType
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This variable indicates the storage type for this object.
The agent MUST ensure that this object’s value remains consistent with the storage type of any rows in other tables that contain pointers to this row.
In particular, the storage type of this row must be at least as permanent as that of any row that point to it.
Conceptual rows having the value ‘permanent’ need not allow write-access to any columnar objects in the row."
REFERENCE
"See RFC2579."
DEFVAL { volatile }
::= { gmplsLabelEntry 18 }

-- End of GMPLS Label Table

-- Module compliance.

gmplsLabelGroups
OBJECT IDENTIFIER ::= { gmplsLabelConformance 1 }

gmplsLabelCompliances
OBJECT IDENTIFIER ::= { gmplsLabelConformance 2 }

gmplsLabelModuleFullCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
"Compliance statement for agents that support the GMPLS Label MIB module."

MODULE -- this module

-- The mandatory groups have to be implemented by LSRs claiming support for this MIB module. This MIB module is, however, not
-- mandatory for a working implementation of a GMPLS LSR with full
-- MIB support if the GMPLS labels in use can be represented within
-- a 32 bit quantity.

MANDATORY-GROUPS {
   gmplsLabelTableGroup
}

Nadeau, Srinivasan, Farrel, Hall and Harrison
-- Units of conformance.

GROUP gmplsLabelTableGroup
DESCRIPTION
"This group is mandatory for devices which support the gmplsLabelTable."

GROUP gmplsLabelPacketGroup
DESCRIPTION
"This group extends gmplsLabelTableGroup for implementations that support packet labels."

GROUP gmplsLabelPortWavelengthGroup
DESCRIPTION
"This group extends gmplsLabelTableGroup for implementations that support port and wavelength labels."

GROUP gmplsLabelFreeformGroup
DESCRIPTION
"This group extends gmplsLabelTableGroup for implementations that support freeform labels."

GROUP gmplsLabelSonetSdhGroup
DESCRIPTION
"This group extends gmplsLabelTableGroup for implementations that support SONET or SDH labels."

GROUP gmplsLabelWavebandGroup
DESCRIPTION
"This group extends gmplsLabelTableGroup for implementations that support Waveband labels."

-- gmplsLabelTable

OBJECT gmplsLabelType
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT gmplsLabelMplsLabel
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT gmplsLabelPortWavelength
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."

OBJECT gmplsLabelFreeform
MIN-ACCESS read-only
DESCRIPTION
"Write access is not required."
OBJECT  gmplsLabelSonetSdhSignalIndex
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelSdhVc
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelSdhVcBranch
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelSonetSdhBranch
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelSonetSdhGroupBranch
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelWavebandId
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelWavebandStart
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelWavebandEnd
MIN-ACCESS  read-only
DESCRIPTION
  "Write access is not required."

OBJECT  gmplsLabelRowStatus
SYNTAX  RowStatus {
    active(1),
    notInService(2)
}
WRITE-SYNTAX RowStatus {
    active(1),
    notInService(2),
    createAndGo(4),
    destroy(6)
}

DESCRIPTION "Support for notInService, createAndWait and notReady
    is not required."

OBJECT  gmplsLabelStorageType
MIN-ACCESS  read-only
DESCRIPTION "Write access is not required."

::= { gmplsLabelCompliances 1 }

-- Units of conformance.

gmplsLabelTableGroup OBJECT-GROUP
OBJECTS {
    gmplsLabelIndexNext,
    gmplsLabelType,
    gmplsLabelRowStatus,
    gmplsLabelStorageType
}
STATUS  current
DESCRIPTION "Necessary, but not sufficient, set of objects to
    implement label table support.  In addition,
    depending on the type of labels supported (for
    example, wavelength labels), the following other
groups defined below are mandatory:
gmplsLabelPacketGroup and/or
gmplsLabelPortWavelengthGroup and/or
gmplsLabelFreeformGroup and/or
gmplsLabelSonetSdhGroup."
::= { gmplsLabelGroups 1 }

gmplsLabelPacketGroup OBJECT-GROUP
OBJECTS {
    gmplsLabelMplsLabel
}
STATUS  current
DESCRIPTION "Object needed to implement Packet (MPLS) labels."
::= { gmplsLabelGroups 2 }

gmplsLabelPortWavelengthGroup OBJECT-GROUP
OBJECTS {
    gmplsLabelPortWavelength

} STATUS current DESCRIPTION "Object needed to implement Port and Wavelength labels." ::= { gmplsLabelGroups 3 }

gmplsLabelFreeformGroup OBJECT-GROUP OBJECTS {
  gmplsLabelFreeform
} STATUS current DESCRIPTION "Object needed to implement Freeform labels." ::= { gmplsLabelGroups 4 }

Nadeau, Srinivasan, Farrel, Hall and Harrison [Page 28]
gmplsLabelSonetSdhGroup OBJECT-GROUP
OBJECTS {
    gmplsLabelSonetSdhSignalIndex,
    gmplsLabelSdhVc,
    gmplsLabelSdhVcBranch,
    gmplsLabelSonetSdhBranch,
    gmplsLabelSonetSdhGroupBranch
}
STATUS current
DESCRIPTION
"Object needed to implement SONET and SDH labels."
::= { gmplsLabelGroups 5 }

gmplsLabelWavebandGroup OBJECT-GROUP
OBJECTS {
    gmplsLabelWavebandId,
    gmplsLabelWavebandStart,
    gmplsLabelWavebandEnd
}
STATUS current
DESCRIPTION
"Object needed to implement Waveband labels."
::= { gmplsLabelGroups 6 }

END

9. Security Considerations

It is clear that the MIB modules described in this document in association with the MPLS-LSR-STD-MIB are potentially useful for monitoring of GMPLS LSRs. These MIB modules can also be used for configuration of certain objects, and anything that can be configured can be incorrectly configured, with potentially disastrous results.

There are a number of management objects defined in these MIB modules with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- the gmplsInterfaceTable, gmplsInSegmentTable, gmplsOutSegmentTable and gmplsLabelTable collectively contain objects to provision GMPLS interfaces, LSPs and their associated parameters on a Label Switching Router (LSR). Unauthorized write access to objects in these tables, could result in disruption of traffic on the network. This is especially true if an LSP has already been established. The use of stronger mechanisms such as SNMPv3 security should be considered where possible. Specifically, SNMPv3 VACM and USM MUST be used with any SNMPv3 agent which implements
these MIB modules.

Some of the readable objects in these MIB modules "i.e., objects with a MAX-ACCESS other than not-accessible" may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly
to even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- the gmplsInterfaceTable, gmplsInSegmentTable, gmplsOutSegmentTable and gmplsLabelTable collectively show the LSP network topology and its capabilities. If an Administrator does not want to reveal this information, then these tables should be considered sensitive/vulnerable.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure "for example by using IPSec", even then, there is no control as to who on the secure network is allowed to access and GET/SET "read/change/create/delete" the objects in these MIB modules. It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework "see [RFC3410], section 8", including full support for the SNMPv3 cryptographic mechanisms "for authentication and privacy".

Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module, is properly configured to give access to the objects only to those principals "users" that have legitimate rights to indeed GET or SET "change/create/delete" them.

10. Acknowledgments

This document extends [LSRMIB]. The authors would like to express their gratitude to all those who worked on that earlier MIB document.

The authors would like to express their thanks to Dan Joyole for his careful review and comments on early versions of the Label Table. Special thanks to Joan Cucchiara and Len Nieman for their help with compilation issues.

11. IANA Considerations

MPLS related standards track MIB modules are rooted under the mplsStdMIB subtree.

One of the MIB modules contained in this document extends tables contained in MPLS MIB modules.

As requested in requested in the GMPLS-TC-STD-MIB [GMPLSTCMIB] the two MIB modules contained in this document should be placed in the
mplsStdMIB subtree as well.

New assignments can only be made via a Standards Action as specified in [RFC2434].

11.1. IANA Considerations for GMPLS-LSR-STD-MIB

The IANA is requested to assign { mplsStdMIB xx } to the GMPLS-LSR-STD-MIB module specified in this document.
11.2. IANA Considerations for GMPLS-LABEL-STD-MIB

The IANA is requested to assign { mplsStdMIB xx } to the GMPLS-LABEL-STD-MIB module specified in this document.

12. References

12.1. Normative References


12.2. Informational References


[RFC3471] Berger, L. (Editor), "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional


13. Authors’ Addresses

Thomas D. Nadeau  
Cisco Systems, Inc.  
300 Apollo Drive  
Chelmsford, MA 01824  
Phone: +1-978-244-3051  
Email: tnadeau@cisco.com

Cheenu Srinivasan  
Bloomberg L.P.  
499 Park Ave.,  
New York, NY 10022  
Phone: +1-212-893-3682  
Email: cheenu@bloomberg.net

Adrian Farrel  
Old Dog Consulting  
Phone: +44-(0)-1978-860944  
Email: adrian@olddog.co.uk

Tim Hall  
Data Connection Ltd.  
100 Church Street  
Enfield, Middlesex, EN2 6BQ, UK  
Phone: +44 20 8366 1177  
Email: tim.hall@dataconnection.com

Ed Harrison  
Data Connection Ltd.  
100 Church Street  
Enfield, Middlesex, EN2 6BQ, UK  
Phone: +44 20 8366 1177  
Email: ed.harrison@dataconnection.com

14. Full Copyright Statement

Copyright (C) The Internet Society (2004). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published
and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for
copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns. This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS 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.

15. Intellectual Property Notice

The IETF takes no position regarding the validity or scope of any intellectual property 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; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11 [RFC2028].

Copies of claims of rights made available for publication 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 Secretariat.

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 practice this standard. Please address the information to the IETF Executive Director.

16. Changes and Pending Work

This section must be removed before the draft progresses to RFC.

16.1. Pending Work

The following work items have been identified for this draft. They will be addressed in a future version.

- Expand conformance statements to give one for monitoring only, and one for monitoring and control.
Determine whether the 'discriminated union' in the Label Table is good MIB.
16.2. Changes from version 3 to version 4
   - Update references.
   - Allow configuration of the Hello timer per interface.
   - Provide support for monitoring technology-specific resources or performance through an arbitrary row pointer.
   - Retire unnecessary gmplsLabelFreeformLength.
   - Update examples.

16.3. Changes from version 2 to version 3
   - Work on basic compilation issues.
   - Provide a next index object to supply the next available arbitrary index into the Label Table.
   - Update references.
   - Update examples.