

Network Working Group
Request for Comments: 4631
Obsoletes: 4327
Category: Standards Track

M. Dubuc
T. Nadeau
Cisco Systems
J. Lang
Sonos, Inc.
E. McGinnis
Hammerhead Systems
A. Farrel
Old Dog Consulting
September 2006

Link Management Protocol (LMP) Management Information Base (MIB)

Status of This Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document provides minor corrections to and obsoletes RFC 4327.

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 the Link Management Protocol (LMP).

Table of Contents

1. The Internet-Standard Management Framework	3
2. Introduction	3
3. Terminology	3
4. Feature Checklist	4
5. Outline	4
6. Brief Description of MIB Objects	5
6.1. lmpNbrTable	5
6.2. lmpControlChannelTable	5
6.3. lmpControlChannelPerfTable	5
6.4. lmpTeLinkTable	5
6.5. lmpLinkVerificationTable	5
6.6. lmpTeLinkPerfTable	6
6.7. lmpDataLinkTable	6
6.8. lmpDataLinkPerfTable	6
7. Example of LMP Control Channel Setup	6
8. Application of the Interfaces Group to LMP	9
8.1. Support of the LMP Layer by ifTable	10
9. LMP MIB Module Definitions	11
10. Security Considerations	78
11. Contributors	79
12. Acknowledgements	79
13. IANA Considerations	79
13.1. IANA Considerations for LMP ifType	79
13.2. IANA Considerations for LMP-MIB	79
14. Changes from RFC 4327 to RFC 4631	79
15. References	80
15.1. Normative References	80
15.2. Informative References	81

1. The Internet-Standard 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].

2. Introduction

Current work is under way in the IETF to specify a suite of protocols to be used as a common control plane and a separate common measurement plane. Generalized MPLS (GMPLS) [RFC3471] and the Link Management Protocol [RFC4204] are key components of this standardization activity. The primary purpose of LMP is to manage traffic engineering (TE) links. Primary goals of LMP are the maintenance of the control channel connectivity, correlation of link properties, verification of data-bearing links, and detection and isolation of link faults.

We describe in this document a MIB module that can be used to manage LMP implementations. This MIB module covers both configuration and performance-monitoring aspects of LMP.

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 [RFC2119].

3. Terminology

This document uses terminology from the document describing the Link Management Protocol [RFC4204]. An "LMP adjacency" is formed between two nodes that support the same capabilities, and LMP messages are exchanged between the node pair over control channels that form this adjacency. Several control channels can be active at the same time. With the exception of messages related to control channel management, anytime an LMP message needs to be transferred to a neighbor node, it can be sent on any of the active control channels. The control channels can also be used to exchange MPLS control plane information or routing information.

LMP is designed to support aggregation of one or more data-bearing links into a traffic-engineering (TE) link. The data-bearing links can be either component links or ports, depending on their multiplexing capability (see [RFC4204] for the distinction between port and component link).

Each TE link is associated with an LMP adjacency, and one or more control channels are used to exchange LMP messages for a particular adjacency. In turn, control channels are used to manage the TE links associated with the LMP adjacency.

4. Feature Checklist

The Link Management Protocol MIB module (LMP-MIB) is designed to satisfy the following requirements and constraints:

- The MIB module supports the enabling and disabling of LMP capability on LMP-capable interfaces of a photonic switch, optical cross-connect, or router.
- The MIB module is used to provide information about LMP adjacencies.
- Support is provided for configuration of the keep-alive and link verification parameters.
- The MIB module is used to express the mapping between local and remote TE links, as well as local and remote interface identifiers for port or component link.
- Performance counters are provided for measuring LMP performance on a per-control channel basis. Performance counters are also provided for measuring LMP performance on the data-bearing links.

Note that the LMP MIB module goes hand-in-hand with the TE Link (TE-LINK-STD-MIB) MIB module [RFC4220]. The TE link table, which is used to associate data-bearing links to TE links, is defined in the TE Link MIB. The TE link table in the LMP MIB module contains TE link information specific to LMP.

5. Outline

Configuring LMP through an optical device involves the following steps:

- Enabling LMP on LMP-capable interfaces through control channel configuration.

- Optionally, specifying link verification parameters.
- Configuring the data-bearing links and associating them to the appropriate TE link (this association is stored in the ifStackTable of the Interfaces Group MIB).

TE links are managed by the control channels that run between the same pair of nodes (LMP adjacency).

6. Brief Description of MIB Objects

Sections 6.1 - 6.8 describe objects pertaining to LMP. The MIB objects were derived from the LMP document [RFC4204].

6.1. lmpNbrTable

The remote node table is used to identify the pair of nodes that exchange LMP messages over control channels.

6.2. lmpControlChannelTable

The control channel table is used for enabling the LMP protocol on LMP-capable interfaces. A photonic switch, optical cross-connect, or router creates an entry in this table for every LMP-capable interface in that device.

6.3. lmpControlChannelPerfTable

The control channel performance table is used for collecting LMP performance counts on a per-control channel basis. Each entry in the lmpControlChannelTable has a corresponding entry in the lmpControlChannelPerfTable.

6.4. lmpTeLinkTable

The TE link table is used for specifying LMP information associated with TE links.

6.5. lmpLinkVerificationTable

The link verification table is used for configuring the LMP link verification parameters of TE links. For every TE link entry in the lmpTeLinkTable that supports the link verification procedure, there is a corresponding entry in the lmpLinkVerificationTable.

6.6. `lmpTeLinkPerfTable`

The TE link performance table is used for collecting LMP performance counts on a per-TE link basis. Each entry in the `lmpTeLinkTable` has a corresponding entry in the `lmpTeLinkPerfTable`.

6.7. `lmpDataLinkTable`

The data-bearing link table is used to specify the data-bearing links that are associated with TE links.

6.8. `lmpDataLinkPerfTable`

The data-bearing link performance table is used for collecting LMP performance counts on data-bearing links.

7. Example of LMP Control Channel Setup

In this section, we provide a brief example of using the MIB objects described in Section 9 to set up an LMP control channel. This example is not meant to illustrate every nuance of the MIB module, but it is intended as an aid to understanding some of the key concepts. It is meant to be read after one goes through the MIB itself.

Suppose that one would like to form an LMP adjacency between two nodes using two control channels. Suppose also that there are three data-bearing links. We also assume that the data-bearing links are ports (lambdas) and that the link verification procedure is not enabled. The following example illustrates which rows and corresponding objects might be created to accomplish this.

First, LMP must be enabled between the pair of nodes.

In `lmpNbrTable`:

```
{
    lmpNbrNodeId                = 'c0000201'H, -- 192.0.2.1
    lmpNbrAdminStatus           = up(1),
    lmpNbrRowStatus              = createAndGo(4),
    lmpNbrStorageType            = nonVolatile(3)
}
```

Then, the control channels must be set up. These are created in the `lmpControlChannelTable`.

In `lmpControlChannelTable`:

```
{
    lmpCcId                      = 1,
```

```

    lmpCcUnderlyingIfIndex      = 1,
    lmpCcIsIf                   = false(2),
    lmpCcAuthentication         = false(2),
    lmpCcHelloInterval         = 15,
    lmpCcHelloIntervalMin      = 15,
    lmpCcHelloIntervalMax      = 1000,
    lmpCcHelloDeadInterval     = 45,
    lmpCcHelloDeadIntervalMin  = 45,
    lmpCcHelloDeadIntervalMax  = 1000,
    lmpCcAdminStatus           = up(1),
    lmpCcRowStatus              = createAndGo(4),
    lmpCcStorageType            = nonVolatile(3)
}

{
    lmpCcId                      = 2,
    lmpCcUnderlyingIfIndex      = 2,
    lmpCcIsIf                   = false(2),
    lmpCcAuthentication         = false(2),
    lmpCcHelloInterval         = 15,
    lmpCcHelloIntervalMin      = 15,
    lmpCcHelloIntervalMax      = 1000,
    lmpCcHelloDeadInterval     = 45,
    lmpCcHelloDeadIntervalMin  = 45,
    lmpCcHelloDeadIntervalMax  = 1000,
    lmpCcAdminStatus           = up(1),
    lmpCcRowStatus              = createAndGo(4),
    lmpCcStorageType            = nonVolatile(3)
}

```

Next, the three data-bearing links are created. For each data-bearing link, an ifEntry with the same ifIndex needs to be created beforehand.

```

    In lmpDataLinkTable:
    {
        ifIndex                  = 41,
        lmpDataLinkAddressType   = unknown(0),
        lmpDataLinkIpAddr       = 'H,
        lmpDataLinkRemoteIpAddress = 'H,
        lmpDataLinkRemoteIfId    = 47,
        lmpDataLinkRowStatus     = createAndGo(4),
        lmpDataLinkStorageType   = nonVolatile(3)
    }

    {
        ifIndex                  = 43,
        lmpDataLinkAddressType   = unknown(0),

```

```

    lmpDataLinkIpAddr          = 'H,
    lmpDataLinkRemoteIpAddress = 'H,
    lmpDataLinkRemoteIfId      = 42,
    lmpDataLinkRowStatus       = createAndGo(4),
    lmpDataLinkStorageType     = nonVolatile(3)
}
{
    ifIndex                    = 44,
    lmpDataLinkAddressType     = unknown(0),
    lmpDataLinkIpAddr          = 'H,
    lmpDataLinkRemoteIpAddress = 'H,
    lmpDataLinkRemoteIfId      = 48,
    lmpDataLinkRowStatus       = createAndGo(4),
    lmpDataLinkStorageType     = nonVolatile(3)
}

```

Note that the data-bearing link type (`lmpDataLinkType`) does not need to be provisioned, as it is automatically populated by the node. The definition of the protection role (primary or secondary) for the data-bearing links is stored in the `componentLinkTable` of the TE Link MIB module [RFC4220].

Then, a TE link is created as an `ifEntry` with `ifType` `teLink` in the `ifTable`.

Once the TE link is created in the `ifTable`, a TE link entry is created in the LMP MIB module to specify TE link information specific to LMP.

```

In lmpTeLinkTable:
{
    ifIndex                = 20,
    lmpTeLinkVerification  = true(1),
    lmpTeLinkFaultManagement = true(1),
    lmpTeLinkDwdm          = false(2),
    lmpTeLinkRowStatus     = createAndGo(4),
    lmpTeLinkStorageType   = nonVolatile(3)
}

```

and in `lmpLinkVerificationTable`:

```

{
    ifIndex                = 20,
    lmpLinkVerifyInterval  = 100,
    lmpLinkVerifyDeadInterval = 300,
    lmpLinkVerifyTransportMechanism = j0Trace(3),
    lmpLinkVerifyAllLinks   = true(1),
    lmpLinkVerifyTransmissionRate = 100000,
    lmpLinkVerifyWavelength = 0,
}

```



```

    lmpLinkVerifyRowStatus      = createAndGo(4),
    lmpLinkVerifyStorageType    = nonVolatile(3)
}

```

The association between the data-bearing links and the TE links is stored in the ifStackTable [RFC2863].

In parallel with the entry created in the lmpTeLinkTable, an entry may be created in the teLinkTable of the TE Link MIB module [RFC4220].

8. Application of the Interfaces Group to LMP

The Interfaces Group [RFC2863] defines generic managed objects for managing interfaces. This memo contains the media-specific extensions to the Interfaces Group for managing LMP control channels that are modeled as interfaces. If the control channel as defined in the lmpControlChannelTable is modeled as an ifEntry, then the following definition applies. An lmpControlChannelTable entry is designated as being represented as an Interfaces MIB ifEntry if the lmpControlChannelEntry object lmpCcIsIf is set to true (1). In this case, the control channel SHOULD be modeled as an ifEntry and provide appropriate interface stacking, as defined below.

This memo assumes the interpretation of the Interfaces Group to be in accordance with [RFC2863], which 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. Since the LMP interface only carries control traffic, it is considered to be below the internetwork layer. Thus, the LMP interface may be represented as an entry in the ifTable. The interrelation of entries in the ifTable is defined by Interfaces Stack Group defined in [RFC2863].

When LMP control channels are modeled as interfaces, the interface stack table must appear as follows for the LMP control channel interfaces:

```

+-----+
| LMP-interface ifType = lmp(227)          +
+-----+
| Underlying Layer...                      +
+-----+

```

In the above diagram, "Underlying Layer..." refers to the ifIndex of any interface type over which the LMP interface will transmit its traffic. Note that if the underlying layer provides multiple access

to its media (i.e., Ethernet), then it is possible to stack multiple LMP interfaces on top of this interface in parallel.

Note that it is not a requirement that LMP control channels be modeled as interfaces. It is acceptable that control channels simply exist as logical connections between adjacent LMP-capable nodes. In this case, `lmpCcIsIf` is set to `false(2)`, and no corresponding entry is made in the `ifTable`.

8.1. Support of the LMP Layer by `ifTable`

Some specific interpretations of `ifTable` for the LMP layer follow.

<code>Object</code>	Use for the LMP layer.
<code>ifIndex</code>	Each LMP interface may be represented by an <code>ifEntry</code> .
<code>ifDescr</code>	Description of the LMP interface.
<code>ifType</code>	The value that is allocated for LMP is 227. This number has been assigned by the IANA.
<code>ifSpeed</code>	The total bandwidth in bits per second for use by the LMP layer.
<code>ifPhysAddress</code>	Unused.
<code>ifAdminStatus</code>	This variable indicates the administrator's intent as to whether LMP should be enabled, disabled, or running in some diagnostic testing mode on this interface. Also see [RFC2863].
<code>ifOperStatus</code>	This value reflects the actual or operational status of LMP on this interface.
<code>ifLastChange</code>	See [RFC2863].
<code>ifInOctets</code>	The number of received octets over the interface; i.e., the number of octets received as LMP packets.
<code>ifOutOctets</code>	The number of transmitted octets over the interface; i.e., the number of octets transmitted as LMP packets.
<code>ifInErrors</code>	The number of LMP packets dropped due to uncorrectable errors.

ifInUnknownProtos The number of received packets discarded during packet header validation, including packets with unrecognized label values.

ifOutErrors See [RFC2863].

ifName Textual name (unique on this system) of the interface or an octet string of zero length.

ifLinkUpDownTrapEnable Default is disabled (2).

ifConnectorPresent Set to false (2).

ifHighSpeed See [RFC2863].

ifHCInOctets The 64-bit version of ifInOctets; supported if required by the compliance statements in [RFC2863].

ifHCOctets The 64-bit version of ifOutOctets; supported if required by the compliance statements in [RFC2863].

ifAlias The nonvolatile 'alias' name for the interface, as specified by a network manager.

ifCounterDiscontinuityTime See [RFC2863].

9. LMP MIB Module Definitions

This MIB module IMPORTS objects from [RFC2578], [RFC2579], [RFC2580], [RFC2863], [RFC4001], and [RFC4220], and it has REFERENCE clauses to [RFC4204], [RFC4207], [RFC4209], [RFC3471], and [RFC2914].

LMP-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE,
transmission, Unsigned32, Counter32, TimeTicks
FROM SNMPv2-SMI -- RFC 2578

MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
FROM SNMPv2-CONF -- RFC 2580

TEXTUAL-CONVENTION, TruthValue, RowStatus, StorageType,
TimeStamp

```
FROM SNMPv2-TC                                -- RFC 2579

InterfaceIndexOrZero, ifIndex
FROM IF-MIB                                    -- RFC 2863

InetAddressType, InetAddress
FROM INET-ADDRESS-MIB                        -- RFC 4001

teLinkRemoteIpAddr, teLinkIncomingIfId, TeLinkEncodingType
FROM TE-LINK-STD-MIB;                        -- RFC 4220

lmpMIB MODULE-IDENTITY
LAST-UPDATED "200608140000Z"  -- 14 August 2006
ORGANIZATION "Common Control and Measurement Protocols (CCAMP)
              Working Group"
CONTACT-INFO
    "
      Martin Dubuc
      Email: dubuc.consulting@sympatico.ca

      Thomas D. Nadeau
      Email: tnadeau@cisco.com

      Jonathan P. Lang
      Email: jplang@ieee.org

      Evan McGinnis
      Email: emcginnis@hammerheadsystems.com

      Adrian Farrel
      Email: adrian@olddog.co.uk"

DESCRIPTION
    "Copyright (C) 2006 The Internet Society. This version of
    the MIB module is part of RFC 4631; see the RFC itself
    for full legal notices.

    This MIB module contains managed object definitions for
    the Link Management Protocol (LMP) as
    defined in 'Link Management Protocol'."

-- Revision history.
REVISION
    "200608140000Z"  -- 14 August 2006
DESCRIPTION
    "Revised version:
    - Fixes textual descriptions of TruthValue settings such that
      True is always 1 and False is always 2.
    - Adds punctuation to REFERENCE clauses."
```

```

    This revision published as RFC 4631"
REVISION
    "200601110000Z"  -- 11 January 2006
DESCRIPTION
    "Initial version published as RFC 4327"
    ::= { transmission 227 }

-- Textual Conventions

LmpInterval ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS      current
    DESCRIPTION
        "The interval delay, in milliseconds."
    SYNTAX      Unsigned32 (1..65535)

LmpRetransmitInterval ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS      current
    DESCRIPTION
        "The retransmission interval delay in milliseconds."
    SYNTAX      Unsigned32 (1..4294967295)

LmpNodeId ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "1d.1d.1d.1d"
    STATUS      current
    DESCRIPTION
        "Represents a Node ID in network byte order.  Node ID is an
        address of type IPv4."
    REFERENCE
        "Section 1.1 of Link Management Protocol, RFC 4204."
    SYNTAX      OCTET STRING(SIZE(4))

-- Top level components of this MIB

-- Notifications
lmpNotifications OBJECT IDENTIFIER ::= { lmpMIB 0 }
-- Tables, Scalars
lmpObjects       OBJECT IDENTIFIER ::= { lmpMIB 1 }
-- Conformance
lmpConformance   OBJECT IDENTIFIER ::= { lmpMIB 2 }

lmpAdminStatus OBJECT-TYPE
    SYNTAX      INTEGER { up(1), down(2) }
    MAX-ACCESS   read-write
    STATUS      current
    DESCRIPTION
        "The desired operational status of LMP on the node.
```

Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

```
DEFVAL      { up }
 ::= { lmpObjects 1 }
```

```
lmpOperStatus OBJECT-TYPE
SYNTAX      INTEGER { up(1), down(2) }
MAX-ACCESS  read-only
STATUS      current
DESCRIPTION
    "The actual operational status of LMP on the node."
 ::= { lmpObjects 2 }
```

-- LMP Neighbor Table

```
lmpNbrTable OBJECT-TYPE
SYNTAX      SEQUENCE OF LmpNbrEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "This table specifies the neighbor node(s) to which control
    channels may be established."
 ::= { lmpObjects 3 }
```

```
lmpNbrEntry OBJECT-TYPE
SYNTAX      LmpNbrEntry
MAX-ACCESS  not-accessible
STATUS      current
DESCRIPTION
    "An entry in this table is created by a LMP-enabled device for
    every pair of nodes that can establish control channels."
INDEX       { lmpNbrNodeId }
 ::= { lmpNbrTable 1 }
```

```
LmpNbrEntry ::= SEQUENCE {
    lmpNbrNodeId          LmpNodeId,
    lmpNbrRetransmitInterval LmpRetransmitInterval,
    lmpNbrRetryLimit      Unsigned32,
    lmpNbrRetransmitDelta  Unsigned32,
    lmpNbrAdminStatus     INTEGER,
    lmpNbrOperStatus      INTEGER,
    lmpNbrRowStatus        RowStatus,
    lmpNbrStorageType      StorageType
}
```

```
lmpNbrNodeId OBJECT-TYPE
SYNTAX      LmpNodeId
```

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This is a unique index for an entry in the LmpNbrTable.

This value represents the remote Node ID."

::= { lmpNbrEntry 1 }

lmpNbrRetransmitInterval OBJECT-TYPE

SYNTAX LmpRetransmitInterval

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object specifies the initial retransmission interval that is used for the retransmission of messages that require acknowledgement. This object, along with lmpNbrRetryLimit, is used to implement the congestion-handling mechanism defined in Section 10 of the Link Management Protocol specification, which is based on RFC 2914."

REFERENCE

"Link Management Protocol, RFC 4204.

Congestion Control Principles, RFC 2914."

DEFVAL { 500 }

::= { lmpNbrEntry 2 }

lmpNbrRetryLimit OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object specifies the maximum number of times a message is transmitted without being acknowledged. A value of 0 is used to indicate that a node should never stop retransmission. This object, along with lmpNbrRetransmitInterval, is used to implement the congestion-handling mechanism as defined in Section 10 of the Link Management Protocol specification, which is based on RFC 2914."

REFERENCE

"Link Management Protocol, RFC 4204.

Congestion Control Principles, RFC 2914."

DEFVAL { 3 }

::= { lmpNbrEntry 3 }

lmpNbrRetransmitDelta OBJECT-TYPE

SYNTAX Unsigned32

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object governs the speed with which the sender increases the retransmission interval, as explained in Section 10 of the Link Management Protocol specification, which is based on RFC 2914. This value is a power used to express the exponential backoff. The ratio of two successive retransmission intervals is (1 + Delta)."

REFERENCE

"Link Management Protocol, RFC 4204.
Congestion Control Principles, RFC 2914."

DEFVAL { 1 }
 ::= { lmpNbrEntry 4 }

lmpNbrAdminStatus OBJECT-TYPE

SYNTAX INTEGER { up(1), down(2) }
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The desired operational status of LMP to this remote node."

::= { lmpNbrEntry 5 }

lmpNbrOperStatus OBJECT-TYPE

SYNTAX INTEGER { up(1), down(2) }
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The actual operational status of LMP to this remote node."

::= { lmpNbrEntry 6 }

lmpNbrRowStatus 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. None of the writable objects in a row can be changed if the status is active(1). All read-create objects must have valid and consistent values before the row can be activated."

::= { lmpNbrEntry 7 }

lmpNbrStorageType OBJECT-TYPE

SYNTAX StorageType
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"The storage type for this conceptual row in the lmpNbrTable. Conceptual rows having the value 'permanent' need not allow write-access to any


```
        columnar object in the row."
DEFVAL      { nonVolatile }
 ::= { lmpNbrEntry 8 }

-- End of lmpNbrTable

lmpCcHelloIntervalDefault OBJECT-TYPE
SYNTAX      LmpInterval
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "This object specifies the default value for the HelloInterval
    parameter used in the Hello protocol keep-alive phase.  It
    indicates how frequently LMP Hello messages will be sent.  It
    is used as the default value for lmpCcHelloInterval.
    Implementations should save the value of this object in
    persistent memory so that it survives restarts or reboot."
REFERENCE
    "Link Management Protocol, RFC 4204."
 ::= { lmpObjects 4 }

lmpCcHelloIntervalDefaultMin OBJECT-TYPE
SYNTAX      LmpInterval
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "This object specifies the default minimum value for the
    HelloInterval parameter.  It is used as a default value
    for lmpCcHelloIntervalMin.  Implementations should save the
    value of this object in persistent memory so that it survives
    restarts or reboot."
 ::= { lmpObjects 5 }

lmpCcHelloIntervalDefaultMax OBJECT-TYPE
SYNTAX      LmpInterval
MAX-ACCESS  read-write
STATUS      current
DESCRIPTION
    "This object specifies the default maximum value for the
    HelloInterval parameter.  It is used as a default value
    for lmpCcHelloIntervalMax.  Implementations should save the
    value of this object in persistent memory so that it survives
    restarts or reboot."
 ::= { lmpObjects 6 }

lmpCcHelloDeadIntervalDefault OBJECT-TYPE
SYNTAX      LmpInterval
MAX-ACCESS  read-write
```

STATUS current

DESCRIPTION

"This object specifies the default HelloDeadInterval parameter to use in the Hello protocol keep-alive phase. It indicates how long a device should wait before declaring the control channel dead. The HelloDeadInterval parameter should be at least three times the value of HelloInterval. It is used as a default value for lmpCcHelloDeadInterval. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpObjects 7 }

lmpCcHelloDeadIntervalDefaultMin OBJECT-TYPE

SYNTAX LmpInterval

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object specifies the default minimum value for the HelloDeadInterval parameter. It is used as a default value for lmpCcHelloDeadIntervalMin. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

::= { lmpObjects 8 }

lmpCcHelloDeadIntervalDefaultMax OBJECT-TYPE

SYNTAX LmpInterval

MAX-ACCESS read-write

STATUS current

DESCRIPTION

"This object specifies the default maximum value for the HelloDeadInterval parameter. It is used as a default value for lmpCcHelloDeadIntervalMax. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

::= { lmpObjects 9 }

-- LMP Control Channel Table

lmpControlChannelTable OBJECT-TYPE

SYNTAX SEQUENCE OF LmpControlChannelEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table specifies LMP control channel information."

::= { lmpObjects 10 }

lmpControlChannelEntry OBJECT-TYPE

SYNTAX LmpControlChannelEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry in this table is created by an LMP-enabled device for every control channel. Whenever a new entry is created with lmpCcIsIf set to true(1), a corresponding entry is created in ifTable as well (see RFC 2863)."

INDEX { lmpCcId }

::= { lmpControlChannelTable 1 }

LmpControlChannelEntry ::= SEQUENCE {

lmpCcId	Unsigned32,
lmpCcUnderlyingIfIndex	InterfaceIndexOrZero,
lmpCcIsIf	TruthValue,
lmpCcNbrNodeId	LmpNodeId,
lmpCcRemoteId	Unsigned32,
lmpCcRemoteAddressType	InetAddressType,
lmpCcRemoteIpAddr	InetAddress,
lmpCcSetupRole	INTEGER,
lmpCcAuthentication	TruthValue,
lmpCcHelloInterval	LmpInterval,
lmpCcHelloIntervalMin	LmpInterval,
lmpCcHelloIntervalMax	LmpInterval,
lmpCcHelloIntervalNegotiated	LmpInterval,
lmpCcHelloDeadInterval	LmpInterval,
lmpCcHelloDeadIntervalMin	LmpInterval,
lmpCcHelloDeadIntervalMax	LmpInterval,
lmpCcHelloDeadIntervalNegotiated	LmpInterval,
lmpCcLastChange	TimeTicks,
lmpCcAdminStatus	INTEGER,
lmpCcOperStatus	INTEGER,
lmpCcRowStatus	RowStatus,
lmpCcStorageType	StorageType

}

lmpCcId OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This value represents the local control channel identifier. The control channel identifier is a non-zero 32-bit number."

::= { lmpControlChannelEntry 1 }

lmpCcUnderlyingIfIndex OBJECT-TYPE

SYNTAX InterfaceIndexOrZero

MAX-ACCESS read-create
 STATUS current
 DESCRIPTION

"If lmpCcIsIf is set to true(1), this object carries the index into the ifTable of the entry that represents the LMP interface over which LMP will transmit its traffic. If this object is set to zero but lmpCcIsIf is set to true(1), the control channel is not currently associated with any underlying interface, and the control channel's operational status must not be up(1); nor should the control channel forward or receive traffic. If lmpCcIsIf is set to false(2), this object should be set to zero and ignored."

::= { lmpControlChannelEntry 2 }

lmpCcIsIf OBJECT-TYPE

SYNTAX TruthValue
 MAX-ACCESS read-create
 STATUS current
 DESCRIPTION

"In implementations where the control channels are modeled as interfaces, the value of this object is true(1), and this control channel is represented by an interface in the interfaces group table as indicated by the value of lmpCcUnderlyingIfIndex. If control channels are not modeled as interfaces, the value of this object is false(2), and there is no corresponding interface for this control channel in the interfaces group table; the value of lmpCcUnderlyingIfIndex should be ignored."

::= { lmpControlChannelEntry 3 }

lmpCcNbrNodeId OBJECT-TYPE

SYNTAX LmpNodeId
 MAX-ACCESS read-create
 STATUS current
 DESCRIPTION

"This is the Node ID of the control channel remote node. This value either is configured or gets created by the node when a Config message is received or when an outgoing Config message is acknowledged by the remote node."

::= { lmpControlChannelEntry 4 }

lmpCcRemoteId OBJECT-TYPE

SYNTAX Unsigned32
 MAX-ACCESS read-only
 STATUS current
 DESCRIPTION

"This value represents the remote control channel identifier (32-bit number). It is determined during the negotiation phase. A value of zero means that the remote control channel identifier has not yet been learned."

::= { lmpControlChannelEntry 5 }

lmpCcRemoteAddressType OBJECT-TYPE

SYNTAX InetAddressType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value represents the remote control channel IP address type. In point-to-point configuration, this value can be set to unknown(0)."

::= { lmpControlChannelEntry 6 }

lmpCcRemoteIpAddr OBJECT-TYPE

SYNTAX InetAddress

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value represents the remote control channel Internet address for numbered control channel. The type of this address is determined by lmpCcRemoteAddressType. The control channel must be numbered on non-point-to-point configuration. For point-to-point configuration, the remote control channel address can be of type unknown, in which case this object must be a zero-length string. The lmpCcRemoteId object then identifies the unnumbered address."

::= { lmpControlChannelEntry 7 }

lmpCcSetupRole OBJECT-TYPE

SYNTAX INTEGER { active(1), passive(2) }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The role that this node should take during establishment of this control channel. An active node will initiate establishment. A passive node will wait for the remote node to initiate. A pair of nodes that both take the passive role will never establish communications."

DEFVAL { active }

::= { lmpControlChannelEntry 8 }

lmpCcAuthentication OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-create

STATUS current
DESCRIPTION
"This object indicates whether the control channel must use authentication."
REFERENCE
"Link Management Protocol, RFC 4204."
 ::= { lmpControlChannelEntry 9 }

lmpCcHelloInterval OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This object specifies the value of the HelloInterval parameter. The default value for this object should be set to lmpCcHelloIntervalDefault."
 ::= { lmpControlChannelEntry 10 }

lmpCcHelloIntervalMin OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This object specifies the minimum value for the HelloInterval parameter. The default value for this object should be set to lmpCcHelloIntervalMinDefault."
 ::= { lmpControlChannelEntry 11 }

lmpCcHelloIntervalMax OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"This object specifies the maximum value for the HelloInterval parameter. The default value for this object should be set to lmpCcHelloIntervalMaxDefault."
 ::= { lmpControlChannelEntry 12 }

lmpCcHelloIntervalNegotiated OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Once the control channel is active, this object represents the negotiated HelloInterval value."
 ::= { lmpControlChannelEntry 13 }

lmpCcHelloDeadInterval OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"This object specifies the value of the HelloDeadInterval parameter. The default value for this object should be set to lmpCcHelloDeadIntervalDefault."

::= { lmpControlChannelEntry 14 }

lmpCcHelloDeadIntervalMin OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"This object specifies the minimum value for the HelloDeadInterval parameter. The default value for this object should be set to lmpCcHelloDeadIntervalMinDefault."

::= { lmpControlChannelEntry 15 }

lmpCcHelloDeadIntervalMax OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"This object specifies the maximum value for the HelloDeadInterval parameter. The default value for this object should be set to lmpCcHelloIntervalMaxDefault."

::= { lmpControlChannelEntry 16 }

lmpCcHelloDeadIntervalNegotiated OBJECT-TYPE

SYNTAX LmpInterval
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"Once the control channel is active, this object represents the negotiated HelloDeadInterval value."

::= { lmpControlChannelEntry 17 }

lmpCcLastChange OBJECT-TYPE

SYNTAX TimeTicks
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"The value of sysUpTime at the time the control channel entered its current operational state. If the current state was entered prior to the last re-initialization of the local network management subsystem, then this object contains a zero value."

```
::= { lmpControlChannelEntry 18 }
```

```
lmpCcAdminStatus OBJECT-TYPE
    SYNTAX      INTEGER { up(1), down(2) }
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The desired operational status of this control channel."
    ::= { lmpControlChannelEntry 19 }
```

```
lmpCcOperStatus OBJECT-TYPE
    SYNTAX      INTEGER {
        up(1),
        down(2),
        configSnd(3),
        configRcv(4),
        active(5),
        goingDown(6)
    }
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The actual operational status of this control channel."
    ::= { lmpControlChannelEntry 20 }
```

```
lmpCcRowStatus 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.  None of the writable objects
        in a row can be changed if the status is active(1).
        All read-create objects must have valid and consistent
        values before the row can be activated."
    ::= { lmpControlChannelEntry 21 }
```

```
lmpCcStorageType OBJECT-TYPE
    SYNTAX      StorageType
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The storage type for this conceptual row in the
        lmpControlChannelTable.  Conceptual rows having the value
        'permanent' need not allow write-access to any
        columnar object in the row."

    DEFVAL      { nonVolatile }
```



```

 ::= { lmpControlChannelEntry 22 }

-- End of lmpControlChannelTable

-- LMP Control Channel Performance Table

lmpControlChannelPerfTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF LmpControlChannelPerfEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This table specifies LMP control channel performance
        counters."
    ::= { lmpObjects 11 }

lmpControlChannelPerfEntry OBJECT-TYPE
    SYNTAX          LmpControlChannelPerfEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in this table is created by a LMP-enabled device for
        every control channel. lmpCcCounterDiscontinuityTime is used
        to indicate potential discontinuity for all counter objects
        in this table."
    INDEX           { lmpCcId }
    ::= { lmpControlChannelPerfTable 1 }

LmpControlChannelPerfEntry ::= SEQUENCE {
    lmpCcInOctets          Counter32,
    lmpCcInDiscards        Counter32,
    lmpCcInErrors          Counter32,
    lmpCcOutOctets         Counter32,
    lmpCcOutDiscards       Counter32,
    lmpCcOutErrors         Counter32,
    lmpCcConfigReceived    Counter32,
    lmpCcConfigSent        Counter32,
    lmpCcConfigRetransmit  Counter32,
    lmpCcConfigAckReceived Counter32,
    lmpCcConfigAckSent     Counter32,
    lmpCcConfigNackReceived Counter32,
    lmpCcConfigNackSent    Counter32,
    lmpCcHelloReceived     Counter32,
    lmpCcHelloSent         Counter32,
    lmpCcBeginVerifyReceived Counter32,
    lmpCcBeginVerifySent   Counter32,
    lmpCcBeginVerifyRetransmit Counter32,
    lmpCcBeginVerifyAckReceived Counter32,

```

```

    lmpCcBeginVerifyAckSent          Counter32,
    lmpCcBeginVerifyNackReceived     Counter32,
    lmpCcBeginVerifyNackSent        Counter32,
    lmpCcEndVerifyReceived           Counter32,
    lmpCcEndVerifySent               Counter32,
    lmpCcEndVerifyRetransmit         Counter32,
    lmpCcEndVerifyAckReceived        Counter32,
    lmpCcEndVerifyAckSent            Counter32,
    lmpCcTestStatusSuccessReceived   Counter32,
    lmpCcTestStatusSuccessSent       Counter32,
    lmpCcTestStatusSuccessRetransmit Counter32,
    lmpCcTestStatusFailureReceived   Counter32,
    lmpCcTestStatusFailureSent       Counter32,
    lmpCcTestStatusFailureRetransmit Counter32,
    lmpCcTestStatusAckReceived        Counter32,
    lmpCcTestStatusAckSent            Counter32,
    lmpCcLinkSummaryReceived          Counter32,
    lmpCcLinkSummarySent              Counter32,
    lmpCcLinkSummaryRetransmit        Counter32,
    lmpCcLinkSummaryAckReceived        Counter32,
    lmpCcLinkSummaryAckSent            Counter32,
    lmpCcLinkSummaryNackReceived       Counter32,
    lmpCcLinkSummaryNackSent          Counter32,
    lmpCcChannelStatusReceived         Counter32,
    lmpCcChannelStatusSent             Counter32,
    lmpCcChannelStatusRetransmit       Counter32,
    lmpCcChannelStatusAckReceived       Counter32,
    lmpCcChannelStatusAckSent           Counter32,
    lmpCcChannelStatusReqReceived       Counter32,
    lmpCcChannelStatusReqSent           Counter32,
    lmpCcChannelStatusReqRetransmit     Counter32,
    lmpCcChannelStatusRspReceived       Counter32,
    lmpCcChannelStatusRspSent           Counter32,
    lmpCcCounterDiscontinuityTime      TimeStamp
}

lmpCcInOctets OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The total number of LMP message octets received on the
         control channel."
    ::= { lmpControlChannelPerfEntry 1 }

lmpCcInDiscards OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only

```

STATUS current
DESCRIPTION
 "The number of inbound packets that were chosen to be discarded even though no errors had been detected. One possible reason for discarding such a packet could be to free up buffer space."
 ::= { lmpControlChannelPerfEntry 2 }

lmpCcInErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of inbound packets that contained errors preventing them from being processed by LMP."
 ::= { lmpControlChannelPerfEntry 3 }

lmpCcOutOctets OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The total number of LMP message octets transmitted out of the control channel."
 ::= { lmpControlChannelPerfEntry 4 }

lmpCcOutDiscards OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of outbound packets that were chosen to be discarded even though no errors had been detected to prevent their being transmitted. One possible reason for discarding such a packet could be to free up buffer space."
 ::= { lmpControlChannelPerfEntry 5 }

lmpCcOutErrors OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "The number of outbound packets that could not be transmitted because of errors."
 ::= { lmpControlChannelPerfEntry 6 }

lmpCcConfigReceived OBJECT-TYPE

```
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "This object counts the number of Config messages that have
    been received on this control channel."
 ::= { lmpControlChannelPerfEntry 7 }
```

```
lmpCcConfigSent OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "This object counts the number of Config messages that have
    been sent on this control channel."
 ::= { lmpControlChannelPerfEntry 8 }
```

```
lmpCcConfigRetransmit OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "This object counts the number of Config messages that
    have been retransmitted over this control channel."
 ::= { lmpControlChannelPerfEntry 9 }
```

```
lmpCcConfigAckReceived OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "This object counts the number of ConfigAck messages that have
    been received on this control channel."
 ::= { lmpControlChannelPerfEntry 10 }
```

```
lmpCcConfigAckSent OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
DESCRIPTION
    "This object counts the number of ConfigAck messages that have
    been sent on this control channel."
 ::= { lmpControlChannelPerfEntry 11 }
```

```
lmpCcConfigNackReceived OBJECT-TYPE
SYNTAX          Counter32
MAX-ACCESS      read-only
STATUS          current
```

DESCRIPTION

"This object counts the number of ConfigAck messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 12 }

lmpCcConfigAckSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ConfigAck messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 13 }

lmpCcHelloReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of Hello messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 14 }

lmpCcHelloSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of Hello messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 15 }

lmpCcBeginVerifyReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of BeginVerify messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 16 }

lmpCcBeginVerifySent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of BeginVerify messages that have been sent on this control channel."

```
::= { lmpControlChannelPerfEntry 17 }
```

```
lmpCcBeginVerifyRetransmit OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"This object counts the number of BeginVerify messages that  
have been retransmitted over this control channel."
```

```
::= { lmpControlChannelPerfEntry 18 }
```

```
lmpCcBeginVerifyAckReceived OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"This object counts the number of BeginVerifyAck messages that  
have been received on this control channel."
```

```
::= { lmpControlChannelPerfEntry 19 }
```

```
lmpCcBeginVerifyAckSent OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"This object counts the number of BeginVerifyAck messages that  
have been sent on this control channel."
```

```
::= { lmpControlChannelPerfEntry 20 }
```

```
lmpCcBeginVerifyNackReceived OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"This object counts the number of BeginVerifyNack messages that  
have been received on this control channel."
```

```
::= { lmpControlChannelPerfEntry 21 }
```

```
lmpCcBeginVerifyNackSent OBJECT-TYPE
```

```
SYNTAX Counter32
```

```
MAX-ACCESS read-only
```

```
STATUS current
```

```
DESCRIPTION
```

```
"This object counts the number of BeginVerifyNack messages that  
have been sent on this control channel."
```

```
::= { lmpControlChannelPerfEntry 22 }
```

```
lmpCcEndVerifyReceived OBJECT-TYPE
```

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object counts the number of EndVerify messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 23 }

lmpCcEndVerifySent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object counts the number of EndVerify messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 24 }

lmpCcEndVerifyRetransmit OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object counts the number of EndVerify messages that have been retransmitted over this control channel."

::= { lmpControlChannelPerfEntry 25 }

lmpCcEndVerifyAckReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object counts the number of EndVerifyAck messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 26 }

lmpCcEndVerifyAckSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"This object counts the number of EndVerifyAck messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 27 }

lmpCcTestStatusSuccessReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of TestStatusSuccess messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 28 }

lmpCcTestStatusSuccessSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusSuccess messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 29 }

lmpCcTestStatusSuccessRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusSuccess messages that have been retransmitted over this control channel."

::= { lmpControlChannelPerfEntry 30 }

lmpCcTestStatusFailureReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusFailure messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 31 }

lmpCcTestStatusFailureSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusFailure messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 32 }

lmpCcTestStatusFailureRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusFailure messages that have been retransmitted over this control channel."


```
::= { lmpControlChannelPerfEntry 33 }
```

lmpCcTestStatusAckReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusAck messages
that have been received on this control channel."

```
::= { lmpControlChannelPerfEntry 34 }
```

lmpCcTestStatusAckSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusAck messages
that have been sent on this control channel."

```
::= { lmpControlChannelPerfEntry 35 }
```

lmpCcLinkSummaryReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages
that have been received on this control channel."

```
::= { lmpControlChannelPerfEntry 36 }
```

lmpCcLinkSummarySent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages
that have been sent on this control channel."

```
::= { lmpControlChannelPerfEntry 37 }
```

lmpCcLinkSummaryRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages that
have been retransmitted over this control channel."

```
::= { lmpControlChannelPerfEntry 38 }
```

lmpCcLinkSummaryAckReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This object counts the number of LinkSummaryAck messages
 that have been received on this control channel."
 ::= { lmpControlChannelPerfEntry 39 }

lmpCcLinkSummaryAckSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This object counts the number of LinkSummaryAck messages
 that have been sent on this control channel."
 ::= { lmpControlChannelPerfEntry 40 }

lmpCcLinkSummaryNackReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This object counts the number of LinkSummaryNack messages
 that have been received on this control channel."
 ::= { lmpControlChannelPerfEntry 41 }

lmpCcLinkSummaryNackSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This object counts the number of LinkSummaryNack messages
 that have been sent on this control channel."
 ::= { lmpControlChannelPerfEntry 42 }

lmpCcChannelStatusReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
 "This object counts the number of ChannelStatus messages
 that have been received on this control channel."
 ::= { lmpControlChannelPerfEntry 43 }

lmpCcChannelStatusSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of ChannelStatus messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 44 }

lmpCcChannelStatusRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatus messages that have been retransmitted on this control channel."

::= { lmpControlChannelPerfEntry 45 }

lmpCcChannelStatusAckReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusAck messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 46 }

lmpCcChannelStatusAckSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatus messages that have been sent on this control channel."

::= { lmpControlChannelPerfEntry 47 }

lmpCcChannelStatusReqReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusRequest messages that have been received on this control channel."

::= { lmpControlChannelPerfEntry 48 }

lmpCcChannelStatusReqSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusRequest messages that have been sent on this control channel."

```
 ::= { lmpControlChannelPerfEntry 49 }

lmpCcChannelStatusReqRetransmit OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object counts the number of ChannelStatusRequest messages
         that have been retransmitted on this control channel."
    ::= { lmpControlChannelPerfEntry 50 }

lmpCcChannelStatusRspReceived OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object counts the number of ChannelStatusResponse messages
         that have been received on this control channel."
    ::= { lmpControlChannelPerfEntry 51 }

lmpCcChannelStatusRspSent OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This object counts the number of ChannelStatusResponse messages
         that have been sent on this control channel."
    ::= { lmpControlChannelPerfEntry 52 }

lmpCcCounterDiscontinuityTime OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "The value of sysUpTime on the most recent occasion at which
         one or more of this control channel's counters suffered a
         discontinuity. The relevant counters are the specific
         instances associated with this control channel of any
         Counter32 object contained in the lmpControlChannelPerfTable.
         If no such discontinuities have occurred since the last re-
         initialization of the local management subsystem, then this
         object contains a zero value."
    ::= { lmpControlChannelPerfEntry 53 }

-- End of lmpControlChannelPerfTable

-- LMP TE Link Table
```

lmpTeLinkTable OBJECT-TYPE

SYNTAX SEQUENCE OF LmpTeLinkEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table specifies the LMP-specific TE link information. Overall TE link information is kept in three separate tables: ifTable for interface-specific information, lmpTeLinkTable for LMP specific information, and teLinkTable for generic TE link information. ifIndex is the common index to all tables."

::= { lmpObjects 12 }

lmpTeLinkEntry OBJECT-TYPE

SYNTAX LmpTeLinkEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry in this table exists for each ifEntry with an ifType of teLink(200) that is managed by LMP. An ifEntry with an ifIndex must exist before the corresponding lmpTeLinkEntry is created. If a TE link entry in the ifTable is destroyed, then so is the corresponding entry in the lmpTeLinkTable. The administrative status value is controlled from the ifEntry. Setting the administrative status to testing prompts LMP to start link verification on the TE link. Information about the TE link that is not LMP specific is contained in the teLinkTable of the TE-LINK-STD-MIB MIB module."

INDEX { ifIndex }

::= { lmpTeLinkTable 1 }

```

LmpTeLinkEntry ::= SEQUENCE {
    lmpTeLinkNbrRemoteNodeId LmpNodeId,
    lmpTeLinkVerification    TruthValue,
    lmpTeLinkFaultManagement TruthValue,
    lmpTeLinkDwdm            TruthValue,
    lmpTeLinkOperStatus      INTEGER,
    lmpTeLinkRowStatus       RowStatus,
    lmpTeLinkStorageType     StorageType
}

```

lmpTeLinkNbrRemoteNodeId OBJECT-TYPE

SYNTAX LmpNodeId

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This is the Node ID of the TE link remote node. This value may be learned during the control channel parameter negotiation

phase (in the Config message). Node ID is an address whose type must be IPv4."
 ::= { lmpTeLinkEntry 1 }

lmpTeLinkVerification OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"This object indicates whether the LMP link verification procedure is enabled for this TE link."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpTeLinkEntry 2 }

lmpTeLinkFaultManagement OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"This object indicates whether the LMP fault management procedure is enabled on this TE link."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpTeLinkEntry 3 }

lmpTeLinkDwdm OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-create
STATUS current
DESCRIPTION

"This object indicates whether the LMP DWDM procedure is enabled on this TE link."

REFERENCE

"Link Management Protocol (LMP) for Dense Wavelength Division Multiplexing (DWDM) Optical Line Systems, RFC 4209."

::= { lmpTeLinkEntry 4 }

lmpTeLinkOperStatus OBJECT-TYPE

SYNTAX INTEGER {
 up(1), down(2), testing(3), init(4), degraded(5)
}
MAX-ACCESS read-only
STATUS current
DESCRIPTION

"The actual operational status of this TE link. The status is set to testing when the TE link is performing link verification. A degraded state indicates that there is

no active control channel between the pair of nodes that form the endpoints of the TE link, but that at least one data-bearing link on the TE link is allocated."

::= { lmpTeLinkEntry 5 }

lmpTeLinkRowStatus 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. None of the writable objects in a row can be changed if the status is active(1). All read-create objects must have valid and consistent values before the row can be activated."

::= { lmpTeLinkEntry 6 }

lmpTeLinkStorageType OBJECT-TYPE

SYNTAX StorageType
MAX-ACCESS read-create
STATUS current

DESCRIPTION

"The storage type for this conceptual row in the lmpTeLinkTable. Conceptual rows having the value 'permanent' need not allow write-access to any columnar object in the row."

DEFVAL { nonVolatile }

::= { lmpTeLinkEntry 7 }

-- End of lmpTeLinkTable

lmpGlobalLinkVerificationInterval OBJECT-TYPE

SYNTAX Unsigned32
UNITS "milliseconds"
MAX-ACCESS read-write
STATUS current

DESCRIPTION

"This object indicates how often the link verification procedure is executed. The interval is in milliseconds. A value of 0 is used to indicate that the link verification procedure should not be executed. The interval specified in this object should be large enough to allow the verification procedure to be completed before the start of the next interval. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

::= { lmpObjects 13 }

-- LMP Link Verification Table

lmpLinkVerificationTable OBJECT-TYPE

SYNTAX SEQUENCE OF LmpLinkVerificationEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"This table specifies TE link information associated with the LMP verification procedure."

::= { lmpObjects 14 }

lmpLinkVerificationEntry OBJECT-TYPE

SYNTAX LmpLinkVerificationEntry

MAX-ACCESS not-accessible

STATUS current

DESCRIPTION

"An entry in this table is created by an LMP-enabled device for every TE link that supports the LMP verification procedure."

INDEX { ifIndex }

::= { lmpLinkVerificationTable 1 }

LmpLinkVerificationEntry ::= SEQUENCE {

lmpLinkVerifyInterval LmpInterval,

lmpLinkVerifyDeadInterval LmpInterval,

lmpLinkVerifyTransportMechanism BITS,

lmpLinkVerifyAllLinks TruthValue,

lmpLinkVerifyTransmissionRate Unsigned32,

lmpLinkVerifyWavelength Unsigned32,

lmpLinkVerifyRowStatus RowStatus,

lmpLinkVerifyStorageType StorageType

}

lmpLinkVerifyInterval OBJECT-TYPE

SYNTAX LmpInterval

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object specifies the VerifyInterval parameter used in the LMP link verification process. It indicates the interval at which the Test messages are sent."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpLinkVerificationEntry 1 }

lmpLinkVerifyDeadInterval OBJECT-TYPE

SYNTAX LmpInterval

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This object specifies the VerifyDeadInterval parameter used in the verification of the physical connectivity of data-bearing links. It specifies the observation period used to detect a Test message at the remote node."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpLinkVerificationEntry 2 }

lmpLinkVerifyTransportMechanism OBJECT-TYPE

SYNTAX BITS {
 -- All encoding types:
 payload(0),
 -- SONET/SDH encoding type:
 dccSectionOverheadBytes(1),
 dccLineOverheadBytes(2),
 j0Trace(3),
 j1Trace(4),
 j2Trace(5)
 }

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This defines the transport mechanism for the Test messages. The scope of this bit mask is restricted to each link encoding type. The local node will set the bits corresponding to the various mechanisms it can support for transmitting LMP Test messages. The receiver chooses the appropriate mechanism in the BeginVerifyAck message."

REFERENCE

"Link Management Protocol, RFC 4204
 Synchronous Optical Network (SONET)/Synchronous Digital Hierarchy (SDH) Encoding for Link Management Protocol (LMP) Test Messages, RFC 4207."

::= { lmpLinkVerificationEntry 3 }

lmpLinkVerifyAllLinks OBJECT-TYPE

SYNTAX TruthValue

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"A value of true(1) for this object indicates that the verification process checks all unallocated links; otherwise, only the new ports or component links that have been added to this TE link are verified."

::= { lmpLinkVerificationEntry 4 }

lmpLinkVerifyTransmissionRate OBJECT-TYPE

SYNTAX Unsigned32

UNITS "bytes per second"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This is the transmission rate of the data link over which the Test messages will be transmitted and is expressed in bytes per second."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpLinkVerificationEntry 5 }

lmpLinkVerifyWavelength OBJECT-TYPE

SYNTAX Unsigned32

UNITS "nanometers"

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"This value corresponds to the wavelength at which the Test messages will be transmitted and is measured in nanometers (nm). If each data-bearing link corresponds to a separate wavelength, then this value should be set to 0."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpLinkVerificationEntry 6 }

lmpLinkVerifyRowStatus 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. None of the writable objects in a row can be changed if the status is active(1). All read-create objects must have valid and consistent values before the row can be activated."

::= { lmpLinkVerificationEntry 7 }

lmpLinkVerifyStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The storage type for this conceptual row in the lmpLinkVerificationTable. Conceptual rows having the value 'permanent' need not allow write-access to any

```

        columnar object in the row."
DEFVAL      { nonVolatile }
 ::= { lmpLinkVerificationEntry 8 }

-- End of lmpLinkVerificationTable

-- LMP TE Link Performance Table

lmpTeLinkPerfTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LmpTeLinkPerfEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "This table specifies LMP TE link performance counters."
    ::= { lmpObjects 15 }

lmpTeLinkPerfEntry OBJECT-TYPE
    SYNTAX      LmpTeLinkPerfEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "An entry in this table is created by an LMP-enabled device for
        every TE link. lmpTeCounterDiscontinuityTime is used
        to indicate potential discontinuity for all counter objects
        in this table."
    INDEX       { ifIndex }
    ::= { lmpTeLinkPerfTable 1 }

LmpTeLinkPerfEntry ::= SEQUENCE {
    lmpTeInOctets          Counter32,
    lmpTeOutOctets         Counter32,
    lmpTeBeginVerifyReceived Counter32,
    lmpTeBeginVerifySent   Counter32,
    lmpTeBeginVerifyRetransmit Counter32,
    lmpTeBeginVerifyAckReceived Counter32,
    lmpTeBeginVerifyAckSent   Counter32,
    lmpTeBeginVerifyNackReceived Counter32,
    lmpTeBeginVerifyNackSent   Counter32,
    lmpTeEndVerifyReceived   Counter32,
    lmpTeEndVerifySent       Counter32,
    lmpTeEndVerifyRetransmit Counter32,
    lmpTeEndVerifyAckReceived Counter32,
    lmpTeEndVerifyAckSent     Counter32,
    lmpTeTestStatusSuccessReceived Counter32,
    lmpTeTestStatusSuccessSent   Counter32,
    lmpTeTestStatusSuccessRetransmit Counter32,
    lmpTeTestStatusFailureReceived Counter32,

```

```

    lmpTeTestStatusFailureSent      Counter32,
    lmpTeTestStatusFailureRetransmit Counter32,
    lmpTeTestStatusAckReceived      Counter32,
    lmpTeTestStatusAckSent          Counter32,
    lmpTeLinkSummaryReceived        Counter32,
    lmpTeLinkSummarySent            Counter32,
    lmpTeLinkSummaryRetransmit      Counter32,
    lmpTeLinkSummaryAckReceived     Counter32,
    lmpTeLinkSummaryAckSent         Counter32,
    lmpTeLinkSummaryNackReceived    Counter32,
    lmpTeLinkSummaryNackSent        Counter32,
    lmpTeChannelStatusReceived      Counter32,
    lmpTeChannelStatusSent          Counter32,
    lmpTeChannelStatusRetransmit    Counter32,
    lmpTeChannelStatusAckReceived   Counter32,
    lmpTeChannelStatusAckSent       Counter32,
    lmpTeChannelStatusReqReceived   Counter32,
    lmpTeChannelStatusReqSent       Counter32,
    lmpTeChannelStatusReqRetransmit Counter32,
    lmpTeChannelStatusRspReceived   Counter32,
    lmpTeChannelStatusRspSent       Counter32,
    lmpTeCounterDiscontinuityTime   TimeStamp
}

lmpTeInOctets OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total number of LMP message octets received for
         this TE link."
    ::= { lmpTeLinkPerfEntry 1 }

lmpTeOutOctets OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "The total number of LMP message octets transmitted out
         for this TE link."
    ::= { lmpTeLinkPerfEntry 2 }

lmpTeBeginVerifyReceived OBJECT-TYPE
    SYNTAX      Counter32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
        "This object counts the number of BeginVerify messages that have

```

been received for this TE link."
 ::= { lmpTeLinkPerfEntry 3 }

lmpTeBeginVerifySent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of BeginVerify messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 4 }

lmpTeBeginVerifyRetransmit OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of BeginVerify messages that have been retransmitted for this TE link."

::= { lmpTeLinkPerfEntry 5 }

lmpTeBeginVerifyAckReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of BeginVerifyAck messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 6 }

lmpTeBeginVerifyAckSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of BeginVerifyAck messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 7 }

lmpTeBeginVerifyNackReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of BeginVerifyNack messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 8 }

lmpTeBeginVerifyNackSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of BeginVerifyNack messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 9 }

lmpTeEndVerifyReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of EndVerify messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 10 }

lmpTeEndVerifySent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of EndVerify messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 11 }

lmpTeEndVerifyRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of EndVerify messages that have been retransmitted over this control channel."

::= { lmpTeLinkPerfEntry 12 }

lmpTeEndVerifyAckReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of EndVerifyAck messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 13 }

lmpTeEndVerifyAckSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current
DESCRIPTION
"This object counts the number of EndVerifyAck messages that
have been sent for this TE link."
::= { lmpTeLinkPerfEntry 14 }

lmpTeTestStatusSuccessReceived OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of TestStatusSuccess messages
that have been received for this TE link."
::= { lmpTeLinkPerfEntry 15 }

lmpTeTestStatusSuccessSent OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of TestStatusSuccess messages
that have been sent for this TE link."
::= { lmpTeLinkPerfEntry 16 }

lmpTeTestStatusSuccessRetransmit OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of TestStatusSuccess messages
that have been retransmitted for this TE link."
::= { lmpTeLinkPerfEntry 17 }

lmpTeTestStatusFailureReceived OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of TestStatusFailure messages
that have been received for this TE link."
::= { lmpTeLinkPerfEntry 18 }

lmpTeTestStatusFailureSent OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of TestStatusFailure messages

that have been sent for this TE link."
 ::= { lmpTeLinkPerfEntry 19 }

lmpTeTestStatusFailureRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusFailure messages
that have been retransmitted on this TE link."

::= { lmpTeLinkPerfEntry 20 }

lmpTeTestStatusAckReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusAck messages that
have been received for this TE link."

::= { lmpTeLinkPerfEntry 21 }

lmpTeTestStatusAckSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of TestStatusAck messages that
have been sent for this TE link."

::= { lmpTeLinkPerfEntry 22 }

lmpTeLinkSummaryReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages that
have been received for this TE link."

::= { lmpTeLinkPerfEntry 23 }

lmpTeLinkSummarySent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages that
have been sent for this TE link."

::= { lmpTeLinkPerfEntry 24 }

lmpTeLinkSummaryRetransmit OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of LinkSummary messages that have been retransmitted over this control channel."

::= { lmpTeLinkPerfEntry 25 }

lmpTeLinkSummaryAckReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of LinkSummaryAck messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 26 }

lmpTeLinkSummaryAckSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of LinkSummaryAck messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 27 }

lmpTeLinkSummaryNackReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of LinkSummaryNack messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 28 }

lmpTeLinkSummaryNackSent OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only
STATUS current

DESCRIPTION

"This object counts the number of LinkSummaryNack messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 29 }

lmpTeChannelStatusReceived OBJECT-TYPE

SYNTAX Counter32
MAX-ACCESS read-only

STATUS current
DESCRIPTION
"This object counts the number of ChannelStatus messages that
have been received for this TE link."
::= { lmpTeLinkPerfEntry 30 }

lmpTeChannelStatusSent OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of ChannelStatus messages that
have been sent for this TE link."
::= { lmpTeLinkPerfEntry 31 }

lmpTeChannelStatusRetransmit OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of ChannelStatus messages that
have been retransmitted for this TE link."
::= { lmpTeLinkPerfEntry 32 }

lmpTeChannelStatusAckReceived OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of ChannelStatusAck messages
that have been received for this TE link."
::= { lmpTeLinkPerfEntry 33 }

lmpTeChannelStatusAckSent OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of ChannelStatus messages
that have been sent for this TE link."
::= { lmpTeLinkPerfEntry 34 }

lmpTeChannelStatusReqReceived OBJECT-TYPE
SYNTAX Counter32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"This object counts the number of ChannelStatusRequest messages

that have been received for this TE link."
 ::= { lmpTeLinkPerfEntry 35 }

lmpTeChannelStatusReqSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusRequest messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 36 }

lmpTeChannelStatusReqRetransmit OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusRequest messages that have been retransmitted for this TE link."

::= { lmpTeLinkPerfEntry 37 }

lmpTeChannelStatusRspReceived OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusResponse messages that have been received for this TE link."

::= { lmpTeLinkPerfEntry 38 }

lmpTeChannelStatusRspSent OBJECT-TYPE

SYNTAX Counter32

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"This object counts the number of ChannelStatusResponse messages that have been sent for this TE link."

::= { lmpTeLinkPerfEntry 39 }

lmpTeCounterDiscontinuityTime OBJECT-TYPE

SYNTAX TimeStamp

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The value of sysUpTime on the most recent occasion at which one or more of this TE link's counters suffered a discontinuity. The relevant counters are the specific instances associated with this TE link of any Counter32

object contained in the lmpTeLinkPerfTable. If no such discontinuities have occurred since the last re-initialization of the local management subsystem, then this object contains a zero value."

```
::= { lmpTeLinkPerfEntry 40 }
```

```
-- End of lmpTeLinkPerfTable
```

```
-- LMP Data Link Table
```

```
lmpDataLinkTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LmpDataLinkEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "This table specifies the data-bearing links managed by the
        LMP."
    ::= { lmpObjects 16 }
```

```
lmpDataLinkEntry OBJECT-TYPE
    SYNTAX      LmpDataLinkEntry
    MAX-ACCESS   not-accessible
    STATUS      current
    DESCRIPTION
        "An entry in this table exists for each ifEntry that represents
        a data-bearing link. An ifEntry with an ifIndex must exist
        before the corresponding lmpDataLinkEntry is created.
        If an entry representing the data-bearing link is destroyed in
        the ifTable, then so is the corresponding entry in the
        lmpDataLinkTable. The administrative status value is
        controlled from the ifEntry. The index to this table is also
        used to get information in the componentLinkTable
        of the TE-LINK-STD-MIB MIB module."
    INDEX       { ifIndex }
    ::= { lmpDataLinkTable 1 }
```

```
LmpDataLinkEntry ::= SEQUENCE {
    lmpDataLinkType          INTEGER,
    lmpDataLinkAddressType   InetAddressType,
    lmpDataLinkIpAddr        InetAddress,
    lmpDataLinkRemoteIpAddress   InetAddress,
    lmpDataLinkRemoteIfId    InterfaceIndexOrZero,
    lmpDataLinkEncodingType  TeLinkEncodingType,
    lmpDataLinkActiveOperStatus  INTEGER,
    lmpDataLinkPassiveOperStatus  INTEGER,
    lmpDataLinkRowStatus      RowStatus,
    lmpDataLinkStorageType    StorageType
}
```

```
}
```

```

lmpDataLinkType OBJECT-TYPE
    SYNTAX      INTEGER {
                    port(1),
                    componentLink(2)
                }
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION
        "This attribute specifies whether this data-bearing link is
        a port or a component link. Component links are multiplex
        capable, whereas ports are not multiplex capable."
    REFERENCE
        "Link Management Protocol, RFC 4204."
    ::= { lmpDataLinkEntry 1 }

```

```

lmpDataLinkAddressType OBJECT-TYPE
    SYNTAX      InetAddressType
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "This attribute specifies the data-bearing link IP address
        type. If the data-bearing link is unnumbered, the address
        type must be set to unknown(0)."
    ::= { lmpDataLinkEntry 2 }

```

```

lmpDataLinkIpAddr OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS   read-create
    STATUS       current
    DESCRIPTION
        "The local Internet address for numbered links. The type
        of this address is determined by the value of
        lmpDataLinkAddressType object.

        For IPv4 and IPv6 numbered links, this object represents
        the local IP address associated with the data-bearing
        link. For an unnumbered link, the local address is
        of type unknown, and this object is set to the zero-length
        string; the ifIndex object then identifies the
        unnumbered address."
    ::= { lmpDataLinkEntry 3 }

```

```

lmpDataLinkRemoteIpAddress OBJECT-TYPE
    SYNTAX      InetAddress
    MAX-ACCESS   read-create
    STATUS       current

```

DESCRIPTION

"The remote Internet address for numbered data-bearing links. The type of this address is determined by the `lmpDataLinkAddressType` object.

For IPv4 and IPv6 numbered links, this object represents the remote IP address associated with the data-bearing link. For an unnumbered link, the remote address is of type unknown, and this object is set to the zero-length string; the `lmpDataLinkRemoteIfId` object then identifies the unnumbered address.

This information is either configured manually or communicated by the remote node during the link verification procedure."

::= { lmpDataLinkEntry 4 }

`lmpDataLinkRemoteIfId` OBJECT-TYPE

SYNTAX InterfaceIndexOrZero

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"Interface identifier of the remote end point. This information is either configured manually or communicated by the remote node during the link verification procedure."

::= { lmpDataLinkEntry 5 }

`lmpDataLinkEncodingType` OBJECT-TYPE

SYNTAX TeLinkEncodingType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The encoding type of the data-bearing link."

REFERENCE

"Generalized MPLS Signaling Functional Description, RFC 3471."

::= { lmpDataLinkEntry 6 }

`lmpDataLinkActiveOperStatus` OBJECT-TYPE

SYNTAX INTEGER {
 upAlloc(1),
 upFree(2),
 down(3),
 testing(4) }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The actual operational status of this data-bearing link

(active FSM)."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpDataLinkEntry 7 }

lmpDataLinkPassiveOperStatus OBJECT-TYPE

SYNTAX INTEGER {
upAlloc(1),
upFree(2),
down(3),
psvTst(4) }

MAX-ACCESS read-only

STATUS current

DESCRIPTION

"The actual operational status of this data-bearing link
(passive FSM)."

REFERENCE

"Link Management Protocol, RFC 4204."

::= { lmpDataLinkEntry 8 }

lmpDataLinkRowStatus 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. None of the writable objects
in a row can be changed if the status is active(1).
All read-create objects must have valid and consistent
values before the row can be activated."

::= { lmpDataLinkEntry 9 }

lmpDataLinkStorageType OBJECT-TYPE

SYNTAX StorageType

MAX-ACCESS read-create

STATUS current

DESCRIPTION

"The storage type for this conceptual row in the
lmpDataLinkTable. Conceptual rows having the value
'permanent' need not allow write-access to any
columnar object in the row."

DEFVAL { nonVolatile }

::= { lmpDataLinkEntry 10 }

-- End of lmpDataLinkTable

-- LMP Data Link Performance Table

```

lmpDataLinkPerfTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF LmpDataLinkPerfEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "This table specifies the data-bearing links LMP performance
        counters."
    ::= { lmpObjects 17 }

lmpDataLinkPerfEntry OBJECT-TYPE
    SYNTAX          LmpDataLinkPerfEntry
    MAX-ACCESS      not-accessible
    STATUS          current
    DESCRIPTION
        "An entry in this table contains information about
        the LMP performance counters for the data-bearing links.
        lmpDataLinkDiscontinuityTime is used to indicate potential
        discontinuity for all counter objects in this table."
    INDEX           { ifIndex }
    ::= { lmpDataLinkPerfTable 1 }

LmpDataLinkPerfEntry ::= SEQUENCE {
    lmpDataLinkTestReceived      Counter32,
    lmpDataLinkTestSent         Counter32,
    lmpDataLinkActiveTestSuccess Counter32,
    lmpDataLinkActiveTestFailure Counter32,
    lmpDataLinkPassiveTestSuccess Counter32,
    lmpDataLinkPassiveTestFailure Counter32,
    lmpDataLinkDiscontinuityTime TimeStamp
}

lmpDataLinkTestReceived OBJECT-TYPE
    SYNTAX          Counter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object counts the number of Test messages that have
        been received on this data-bearing link."
    ::= { lmpDataLinkPerfEntry 1 }

lmpDataLinkTestSent OBJECT-TYPE
    SYNTAX          Counter32
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "This object counts the number of Test messages that have
        been sent on this data-bearing link."
    ::= { lmpDataLinkPerfEntry 2 }

```


`lmpDataLinkActiveTestSuccess OBJECT-TYPE``SYNTAX Counter32``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"This object counts the number of data-bearing link tests that were successful on the active side of this data-bearing link."

`::= { lmpDataLinkPerfEntry 3 }``lmpDataLinkActiveTestFailure OBJECT-TYPE``SYNTAX Counter32``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"This object counts the number of data-bearing link tests that failed on the active side of this data-bearing link."

`::= { lmpDataLinkPerfEntry 4 }``lmpDataLinkPassiveTestSuccess OBJECT-TYPE``SYNTAX Counter32``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"This object counts the number of data-bearing link tests that were successful on the passive side of this data-bearing link."

`::= { lmpDataLinkPerfEntry 5 }``lmpDataLinkPassiveTestFailure OBJECT-TYPE``SYNTAX Counter32``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"This object counts the number of data-bearing link tests that failed on the passive side of this data-bearing link."

`::= { lmpDataLinkPerfEntry 6 }``lmpDataLinkDiscontinuityTime OBJECT-TYPE``SYNTAX TimeStamp``MAX-ACCESS read-only``STATUS current``DESCRIPTION`

"The value of sysUpTime on the most recent occasion at which one or more of this data-bearing link's counters suffered a discontinuity. The relevant counters are the specific instances associated with this data-bearing link of any Counter32 object contained in the lmpDataLinkPerfTable. If

```
    no such discontinuities have occurred since the last re-
    initialization of the local management subsystem, then this
    object contains a zero value."
 ::= { lmpDataLinkPerfEntry 7 }
```

```
-- End of lmpDataLinkPerfTable
```

```
-- Notification Configuration
```

```
lmpNotificationMaxRate OBJECT-TYPE
```

```
SYNTAX      Unsigned32
```

```
MAX-ACCESS  read-write
```

```
STATUS      current
```

```
DESCRIPTION
```

```
"The LMP notification rate depends on the size of the network,
the type of links, the network configuration, the
reliability of the network, etc."
```

When this MIB was designed, care was taken to minimize the amount of notifications generated for LMP purposes. Wherever possible, notifications are state driven, meaning that the notifications are sent only when the system changes state. The only notifications that are repeated and that could cause a problem as far as congestion is concerned are the ones associated with data link verification.

Without any considerations to handling of these notifications, a problem may arise if the number of data links is high. Since the data link verification notifications can happen only once per data link per link verification interval, the notification rate should be sustainable if one chooses an appropriate link verification interval for a given network configuration. For instance, a network of 100 nodes with 5 links of 128 wavelengths each and a link verification of 1 minute, where no more than 10% of the links failed at any given time, would have 1 notification per second sent from each node, or 100 notifications per second for the whole network. The rest of the notifications are negligible compared to this number.

To alleviate the congestion problem, the lmpNotificationMaxRate object can be used to implement a throttling mechanism. It is also possible to enable/disable certain type of notifications.

This variable indicates the maximum number of notifications issued per minute. If events occur more rapidly, the implementation may simply fail to

emit these notifications during that period or may queue them until an appropriate time. A value of 0 means that no throttling is applied and events may be notified at the rate at which they occur. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

::= { lmpObjects 18 }

lmpLinkPropertyNotificationsEnabled OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-write
STATUS current

DESCRIPTION

"If this object is true(1), then it enables the generation of lmpTeLinkPropertyMismatch and lmpDataLinkPropertyMismatch notifications; otherwise, these notifications are not emitted. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

DEFVAL { false }

::= { lmpObjects 19 }

lmpUnprotectedNotificationsEnabled OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-write
STATUS current

DESCRIPTION

"If this object is true(1), then it enables the generation of lmpUnprotected notifications; otherwise, these notifications are not emitted. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

DEFVAL { false }

::= { lmpObjects 20 }

lmpCcUpDownNotificationsEnabled OBJECT-TYPE

SYNTAX TruthValue
MAX-ACCESS read-write
STATUS current

DESCRIPTION

"If this object is true(1), then it enables the generation of lmpControlChannelUp and lmpControlChannelDown notifications; otherwise, these notifications are not emitted. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

DEFVAL { false }

::= { lmpObjects 21 }

lmpTeLinkNotificationsEnabled OBJECT-TYPE

SYNTAX TruthValue
 MAX-ACCESS read-write
 STATUS current

DESCRIPTION

"If this object is true(1), then it enables the generation of lmpTeLinkDegraded and lmpTeLinkNotDegraded notifications; otherwise, these notifications are not emitted. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

DEFVAL { false }
 ::= { lmpObjects 22 }

lmpDataLinkNotificationsEnabled OBJECT-TYPE

SYNTAX TruthValue
 MAX-ACCESS read-write
 STATUS current

DESCRIPTION

"If this object is true(1), then it enables the generation of lmpDataLinkVerificationFailure notification; otherwise, these notifications are not emitted. Implementations should save the value of this object in persistent memory so that it survives restarts or reboot."

DEFVAL { false }
 ::= { lmpObjects 23 }

-- Notifications

-- Link Property Mismatch Notifications

lmpTeLinkPropertyMismatch NOTIFICATION-TYPE

OBJECTS { teLinkRemoteIpAddr,
 teLinkIncomingIfId }

STATUS current

DESCRIPTION

"This notification is generated when a TE link property mismatch is detected on the node. The received remote TE link ID of the misconfigured TE link is represented by either teLinkRemoteIpAddr or teLinkIncomingIfId, depending on whether the TE link is numbered or unnumbered. This notification should not be sent unless lmpLinkPropertyNotificationsEnabled is true(1). It is recommended that this notification be reported only the first time a mismatch is detected. Otherwise, for a given TE link, this notification can occur no more than once per verification interval (lmpGlobalLinkVerificationInterval)."

::= { lmpNotifications 1 }

```
lmpDataLinkPropertyMismatch NOTIFICATION-TYPE
  OBJECTS      { lmpDataLinkType, lmpDataLinkRemoteIfId }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a data-bearing link
    property mismatch is detected on the node. lmpDataLinkType
    is used to identify the local identifiers associated with
    the data link. (The data link interface index can be used
    to determine the TE link interface index, as this
    relationship is captured in the interface stack table.)
    The remote entity interface ID is the remote entity
    interface ID received in the LinkSummary message.
    This notification should not be sent unless
    lmpLinkPropertyNotificationsEnabled is true(1). It is
    recommended that this notification be reported only the
    first time a mismatch is detected. Otherwise, for a given
    data link, this notification can occur no more than once
    per verification interval (lmpGlobalLinkVerificationInterval)."
    ::= { lmpNotifications 2 }

-- Neighbor Notification

lmpUnprotected NOTIFICATION-TYPE
  OBJECTS      { lmpCcNbrNodeId }
  STATUS       current
  DESCRIPTION
    "This notification is generated when there is more than one
    control channel between LMP neighbors and the last redundant
    control channel has failed. If the remaining operational
    control channel fails, then there will be no more control
    channels between the pair of nodes and all the TE links
    between the pair of nodes, will go to degraded state. This
    notification should not be sent unless
    lmpUnprotectedNotificationsEnabled is set to true(1)."
    ::= { lmpNotifications 3 }

-- Control Channel Notifications

lmpControlChannelUp NOTIFICATION-TYPE
  OBJECTS      { lmpCcAdminStatus, lmpCcOperStatus }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a control
    channel transitions to the up operational state. This
    notification should not be sent unless
    lmpCcUpDownNotificationsEnabled is true(1)."
    ::= { lmpNotifications 4 }
```

```
lmpControlChannelDown NOTIFICATION-TYPE
  OBJECTS      { lmpCcAdminStatus, lmpCcOperStatus }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a control channel
    transitions out of the up operational state. This
    notification should not be sent unless
    lmpCcUpDownNotificationsEnabled is true(1)."
```

-- TE Link Notification

```
lmpTeLinkDegraded NOTIFICATION-TYPE
  OBJECTS      { lmpTeLinkOperStatus }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a lmpTeLinkOperStatus
    object for a TE link enters the degraded state. This
    notification should not be sent unless
    lmpTeLinkNotificationsEnabled is true(1)."
```

-- Data-bearing Link Notification

```
lmpTeLinkNotDegraded NOTIFICATION-TYPE
  OBJECTS      { lmpTeLinkOperStatus }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a lmpTeLinkOperStatus
    object for a TE link leaves the degraded state. This
    notification should not be sent unless
    lmpTeLinkNotificationsEnabled is true(1)."
```

-- End of notifications

```
lmpDataLinkVerificationFailure NOTIFICATION-TYPE
  OBJECTS      { lmpDataLinkActiveOperStatus,
                 lmpDataLinkPassiveOperStatus }
  STATUS       current
  DESCRIPTION
    "This notification is generated when a data-bearing
    link verification fails. This notification should not be sent
    unless lmpDataLinkNotificationsEnabled is true(1). For a given
    data link, this notification can occur no more than once per
    verification interval (lmpGlobalLinkVerificationInterval)."
```

```
-- Module compliance

lmpCompliances
  OBJECT IDENTIFIER ::= { lmpConformance 1 }

lmpGroups
  OBJECT IDENTIFIER ::= { lmpConformance 2 }

lmpModuleFullCompliance MODULE-COMPLIANCE
  STATUS current
  DESCRIPTION
    "Compliance statement for agents that support the
     configuration and monitoring of LMP MIB."
  MODULE -- this module

    MANDATORY-GROUPS      { lmpNodeGroup,
                             lmpControlChannelGroup,
                             lmpLinkPropertyCorrelationGroup,
                             lmpPerfGroup,
                             lmpTeLinkGroup,
                             lmpDataLinkGroup }

  GROUP lmpCcIsNotInterfaceGroup
  DESCRIPTION
    "This group is mandatory for devices that support
     control channels that are not interfaces, in addition to
     lmpControlChannelGroup. The following constraint applies:
     lmpCcIsIf must at least be read-only, returning false(2)."
```

```
  GROUP lmpCcIsInterfaceGroup
  DESCRIPTION
    "This group is mandatory for devices that support
     control channels that are interfaces, in addition to
     lmpControlChannelGroup. The following constraint applies:
     lmpCcIsIf must at least be read-only, returning true(1)."
```

```
  GROUP lmpLinkVerificationGroup
  DESCRIPTION
    "This group is mandatory for devices that support
     the link verification procedure."
```

```
  GROUP lmpNotificationGroup
  DESCRIPTION
    "This group is optional."
```

```
-- lmpNbrTable

OBJECT      lmpNbrRowStatus
```

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

DESCRIPTION
    "Support for notReady(3) and createAndWait(5) is
    not required."

-- lmpControlChannelTable

OBJECT      lmpCcRemoteAddressType
SYNTAX      INTEGER { unknown(0), ipv4(1), ipv6(2) }
DESCRIPTION
    "Only ipv4(1) and ipv6(2) address types need to be
    supported for non-point-to-point configurations."

OBJECT      lmpCcRemoteIpAddr
SYNTAX      InetAddress (SIZE(0|4|16))
DESCRIPTION
    "The size of the IP address depends on the address type."

OBJECT      lmpCcRowStatus
SYNTAX      RowStatus { active(1), notInService(2) }
WRITE-SYNTAX RowStatus { active(1), notInService(2),
                        createAndGo(4), destroy(6) }
DESCRIPTION
    "Support for notReady(3) and createAndWait(5) is
    not required."

OBJECT      lmpCcOperStatus
SYNTAX      INTEGER { up(1), down(2) }
DESCRIPTION
    "A value of configSnd(3), configRcv(4), active(5), or
    goingDown(6) need not be supported."

-- lmpTeLinkTable

OBJECT      lmpTeLinkOperStatus
SYNTAX      INTEGER { up(1), down(2), degraded(5) }
DESCRIPTION
    "The testing(3) and init(4) state need not be supported."

OBJECT      lmpTeLinkRowStatus
SYNTAX      RowStatus { active(1), notInService(2) }
WRITE-SYNTAX RowStatus { active(1), notInService(2),
                        createAndGo(4), destroy(6) }
DESCRIPTION
    "Support for notReady(3) and createAndWait(5) is
    not required."
```



```

-- lmpDataLinkTable

OBJECT      lmpDataLinkActiveOperStatus
SYNTAX      INTEGER { upAlloc(1), upFree(2), down(3) }
DESCRIPTION
    "A value of testing(4) need not be supported."
OBJECT      lmpDataLinkPassiveOperStatus
SYNTAX      INTEGER { upAlloc(1), upFree(2), down(3) }
DESCRIPTION
    "A value of psvTst(4) need not be supported."

OBJECT      lmpDataLinkRowStatus
SYNTAX      RowStatus { active(1), notInService(2) }
WRITE-SYNTAX RowStatus { active(1), notInService(2),
                        createAndGo(4), destroy(6) }
DESCRIPTION
    "Support for notReady(3) and createAndWait(5) is
    not required."

 ::= { lmpCompliances 1 }

lmpModuleReadOnlyCompliance MODULE-COMPLIANCE
STATUS current
DESCRIPTION
    "Compliance statement for agents that support the
    monitoring of the LMP MIB."
MODULE -- this module

-- The mandatory groups have to be implemented
-- by all LMP-enabled devices.  However, they may all be supported
-- as read-only objects in the case where manual
-- configuration is not supported.

MANDATORY-GROUPS      { lmpNodeGroup,
                        lmpControlChannelGroup,
                        lmpLinkPropertyCorrelationGroup,
                        lmpPerfGroup,
                        lmpTeLinkGroup,
                        lmpDataLinkGroup }

GROUP lmpCcIsNotInterfaceGroup
DESCRIPTION
    "This group is mandatory for devices that support
    control channels that are not interfaces, in addition to
    lmpControlChannelGroup.  The following constraint applies:
    lmpCcIsIf must at least be read-only, returning false(2)."
```

GROUP lmpCcIsInterfaceGroup

DESCRIPTION

"This group is mandatory for devices that support control channels that are interfaces, in addition to lmpControlChannelGroup. The following constraint applies: lmpCcIsIf must at least be read-only, returning true(1)."

GROUP lmpLinkVerificationGroup

DESCRIPTION

"This group is mandatory for devices that support the link verification procedure."

GROUP lmpNotificationGroup

DESCRIPTION

"This group is optional."

-- Scalars

OBJECT lmpAdminStatus

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpGlobalLinkVerificationInterval

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloIntervalDefault

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloIntervalDefaultMin

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloIntervalDefaultMax

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloDeadIntervalDefault

MIN-ACCESS read-only

DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloDeadIntervalDefaultMin

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

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

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

-- lmpNbrTable

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

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

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

OBJECT      lmpNbrRowStatus
SYNTAX      RowStatus { active(1) }
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required, and active(1) is the
    only status that needs to be supported."

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

-- lmpControlChannelTable

OBJECT      lmpCcUnderlyingIfIndex
MIN-ACCESS  read-only
DESCRIPTION
```

"Write access is not required."

OBJECT lmpCcIsIf
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcNbrNodeId
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcRemoteAddressType
SYNTAX INTEGER { unknown(0), ipv4(1), ipv6(2) }
MIN-ACCESS read-only
DESCRIPTION

"Only ipv4(1) and ipv6(2) address types need to be supported for non-point-to-point configurations."

OBJECT lmpCcRemoteIpAddr
SYNTAX InetAddress (SIZE(0|4|16))
MIN-ACCESS read-only
DESCRIPTION

"The size of the IP address depends on the address type."

OBJECT lmpCcSetupRole
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcAuthentication
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloIntervalMin
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloIntervalMax
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloDeadIntervalMin
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcHelloDeadIntervalMax
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpCcRowStatus
SYNTAX RowStatus { active(1) }
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required, and active(1) is the
only status that needs to be supported."

OBJECT lmpCcOperStatus
SYNTAX INTEGER { up(1), down(2) }
DESCRIPTION

"A value of configSnd(3), configRcv(4), active(5), or
goingDown(6) need not be supported."

OBJECT lmpCcStorageType
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

-- lmpLinkVerificationTable

OBJECT lmpLinkVerifyInterval
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpLinkVerifyDeadInterval
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

OBJECT lmpLinkVerifyAllLinks
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required."

-- lmpTeLinkTable

OBJECT lmpTeLinkNbrRemoteNodeId
MIN-ACCESS read-only
DESCRIPTION

"Write access is not required if the link verification

procedure is enabled."

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

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

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

OBJECT lmpTeLinkOperStatus
SYNTAX INTEGER { up(1), down(2), degraded(5) }
DESCRIPTION
 "The testing(3) and init(4) state need not be supported."

OBJECT lmpTeLinkRowStatus
SYNTAX RowStatus { active(1) }
MIN-ACCESS read-only
DESCRIPTION
 "Write access is not required, and active(1) is the
 only status that needs to be supported."

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

-- lmpTeLinkVerificationTable

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

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

OBJECT lmpLinkVerifyRowStatus
SYNTAX RowStatus { active(1) }

MIN-ACCESS read-only
DESCRIPTION
 "Write access is not required, and active(1) is the
 only status that needs to be supported."

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

-- lmpDataLinkTable

OBJECT lmpDataLinkAddressType
SYNTAX INTEGER { unknown(0), ipv4(1), ipv6(2) }
MIN-ACCESS read-only
DESCRIPTION
 "Only ipv4(1) and ipv6(2) address types need to be
 supported for numbered links. For unnumbered links, the
 unknown(0) address type needs to be supported."

OBJECT lmpDataLinkIpAddr
SYNTAX InetAddress (SIZE(0|4|16))
MIN-ACCESS read-only
DESCRIPTION
 "The size of the data-bearing link IP address depends on
 the type of data-bearing link. Data-bearing link IP
 address size is zero if the link is unnumbered, four if
 the link IP address is IPv4, and sixteen if the link IP
 address is IPv6."

OBJECT lmpDataLinkRemoteIpAddress
SYNTAX InetAddress (SIZE(0|4|16))
MIN-ACCESS read-only
DESCRIPTION
 "Write access is not required if the link verification
 procedure is enabled."

OBJECT lmpDataLinkRemoteIfId
MIN-ACCESS read-only
DESCRIPTION
 "Write access is not required if the link verification
 procedure is enabled."

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

```

OBJECT      lmpDataLinkActiveOperStatus
SYNTAX      INTEGER { upAlloc(1), upFree(2), down(3) }
DESCRIPTION
    "A value of testing(4) need not be supported."

```

```

OBJECT      lmpDataLinkPassiveOperStatus
SYNTAX      INTEGER { upAlloc(1), upFree(2), down(3) }
DESCRIPTION
    "A value of psvTst(4) need not be supported."

```

```

OBJECT      lmpDataLinkRowStatus
SYNTAX      RowStatus { active(1) }
MIN-ACCESS  read-only
DESCRIPTION
    "Write access is not required, and active(1) is the
    only status that needs to be supported."

```

```

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

```

```
 ::= { lmpCompliances 2 }
```

```
-- Units of conformance
```

```

lmpNodeGroup OBJECT-GROUP
  OBJECTS { lmpAdminStatus,
            lmpOperStatus,
            lmpNbrAdminStatus,
            lmpNbrOperStatus,
            lmpNbrRowStatus,
            lmpNbrStorageType,
            lmpUnprotectedNotificationsEnabled,
            lmpNotificationMaxRate
          }
  STATUS   current
  DESCRIPTION
    "Collection of objects that represent LMP node
    configuration."
 ::= { lmpGroups 1 }

```

```

lmpControlChannelGroup OBJECT-GROUP
  OBJECTS {
    lmpNbrRetransmitInterval,
    lmpNbrRetryLimit,
    lmpNbrRetransmitDelta,
    lmpNbrAdminStatus,

```



```

    lmpNbrOperStatus,
    lmpNbrRowStatus,
    lmpNbrStorageType,
    lmpCcHelloIntervalDefault,
    lmpCcHelloIntervalDefaultMin,
    lmpCcHelloIntervalDefaultMax,
    lmpCcHelloDeadIntervalDefault,
    lmpCcHelloDeadIntervalDefaultMin,
    lmpCcHelloDeadIntervalDefaultMax,
    lmpCcNbrNodeId,
    lmpCcRemoteId,
    lmpCcRemoteAddressType,
    lmpCcRemoteIpAddr,
    lmpCcSetupRole,
    lmpCcAuthentication,
    lmpCcHelloInterval,
    lmpCcHelloIntervalMin,
    lmpCcHelloIntervalMax,
    lmpCcHelloIntervalNegotiated,
    lmpCcHelloDeadInterval,
    lmpCcHelloDeadIntervalMin,
    lmpCcHelloDeadIntervalMax,
    lmpCcHelloDeadIntervalNegotiated,
    lmpCcOperStatus,
    lmpCcRowStatus,
    lmpCcStorageType,
    lmpCcUpDownNotificationsEnabled
  }
STATUS    current
DESCRIPTION
    "Objects that can be used to configure LMP interface."
 ::= { lmpGroups 2 }

lmpCcIsInterfaceGroup OBJECT-GROUP
OBJECTS { lmpCcIsIf }
STATUS    current
DESCRIPTION
    "Objects that can be used to configure control channels
     that are interfaces."
 ::= { lmpGroups 3 }

lmpCcIsNotInterfaceGroup OBJECT-GROUP
OBJECTS { lmpCcUnderlyingIfIndex,
          lmpCcIsIf,
          lmpCcLastChange,
          lmpCcAdminStatus
        }
STATUS    current

```

DESCRIPTION

"Objects that can be used to configure control channels
that are not interfaces."

::= { lmpGroups 4 }

lmpLinkPropertyCorrelationGroup OBJECT-GROUP

OBJECTS { lmpLinkPropertyNotificationsEnabled }

STATUS current

DESCRIPTION

"Collection of objects used to configure the link
property correlation procedure."

::= { lmpGroups 5 }

lmpLinkVerificationGroup OBJECT-GROUP

OBJECTS { lmpGlobalLinkVerificationInterval,
lmpLinkVerifyInterval,
lmpLinkVerifyDeadInterval,
lmpLinkVerifyTransportMechanism,
lmpLinkVerifyAllLinks,
lmpLinkVerifyTransmissionRate,
lmpLinkVerifyWavelength,
lmpLinkVerifyRowStatus,
lmpLinkVerifyStorageType,
lmpDataLinkNotificationsEnabled
}

STATUS current

DESCRIPTION

"Collection of objects that represent the link
verification procedure configuration."

::= { lmpGroups 6 }

lmpPerfGroup OBJECT-GROUP

OBJECTS { lmpCcInOctets,
lmpCcInDiscards,
lmpCcInErrors,
lmpCcOutOctets,
lmpCcOutDiscards,
lmpCcOutErrors,
lmpCcConfigReceived,
lmpCcConfigSent,
lmpCcConfigRetransmit,
lmpCcConfigAckReceived,
lmpCcConfigAckSent,
lmpCcConfigNackSent,
lmpCcConfigNackReceived,
lmpCcHelloReceived,
lmpCcHelloSent,
lmpCcBeginVerifyReceived,

lmpCcBeginVerifySent,
lmpCcBeginVerifyRetransmit,
lmpCcBeginVerifyAckReceived,
lmpCcBeginVerifyAckSent,
lmpCcBeginVerifyNackReceived,
lmpCcBeginVerifyNackSent,
lmpCcEndVerifyReceived,
lmpCcEndVerifySent,
lmpCcEndVerifyRetransmit,
lmpCcEndVerifyAckReceived,
lmpCcEndVerifyAckSent,
lmpCcTestStatusSuccessReceived,
lmpCcTestStatusSuccessSent,
lmpCcTestStatusSuccessRetransmit,
lmpCcTestStatusFailureReceived,
lmpCcTestStatusFailureSent,
lmpCcTestStatusFailureRetransmit,
lmpCcTestStatusAckReceived,
lmpCcTestStatusAckSent,
lmpCcLinkSummaryReceived,
lmpCcLinkSummarySent,
lmpCcLinkSummaryRetransmit,
lmpCcLinkSummaryAckReceived,
lmpCcLinkSummaryAckSent,
lmpCcLinkSummaryNackReceived,
lmpCcLinkSummaryNackSent,
lmpCcChannelStatusReceived,
lmpCcChannelStatusSent,
lmpCcChannelStatusRetransmit,
lmpCcChannelStatusAckReceived,
lmpCcChannelStatusAckSent,
lmpCcChannelStatusReqReceived,
lmpCcChannelStatusReqSent,
lmpCcChannelStatusReqRetransmit,
lmpCcChannelStatusRspReceived,
lmpCcChannelStatusRspSent,
lmpCcCounterDiscontinuityTime,
lmpTeInOctets,
lmpTeOutOctets,
lmpTeBeginVerifyReceived,
lmpTeBeginVerifySent,
lmpTeBeginVerifyRetransmit,
lmpTeBeginVerifyAckReceived,
lmpTeBeginVerifyAckSent,
lmpTeBeginVerifyNackReceived,
lmpTeBeginVerifyNackSent,
lmpTeEndVerifyReceived,
lmpTeEndVerifySent,

```

    lmpTeEndVerifyRetransmit,
    lmpTeEndVerifyAckReceived,
    lmpTeEndVerifyAckSent,
    lmpTeTestStatusSuccessReceived,
    lmpTeTestStatusSuccessSent,
    lmpTeTestStatusSuccessRetransmit,
    lmpTeTestStatusFailureReceived,
    lmpTeTestStatusFailureSent,
    lmpTeTestStatusFailureRetransmit,
    lmpTeTestStatusAckReceived,
    lmpTeTestStatusAckSent,
    lmpTeLinkSummaryReceived,
    lmpTeLinkSummarySent,
    lmpTeLinkSummaryRetransmit,
    lmpTeLinkSummaryAckReceived,
    lmpTeLinkSummaryAckSent,
    lmpTeLinkSummaryNackReceived,
    lmpTeLinkSummaryNackSent,
    lmpTeChannelStatusReceived,
    lmpTeChannelStatusSent,
    lmpTeChannelStatusRetransmit,
    lmpTeChannelStatusAckReceived,
    lmpTeChannelStatusAckSent,
    lmpTeChannelStatusReqReceived,
    lmpTeChannelStatusReqSent,
    lmpTeChannelStatusReqRetransmit,
    lmpTeChannelStatusRspSent,
    lmpTeChannelStatusRspReceived,
    lmpTeCounterDiscontinuityTime,
    lmpDataLinkTestReceived,
    lmpDataLinkTestSent,
    lmpDataLinkActiveTestSuccess,
    lmpDataLinkActiveTestFailure,
    lmpDataLinkPassiveTestSuccess,
    lmpDataLinkPassiveTestFailure,
    lmpDataLinkDiscontinuityTime
  }

```

STATUS current

DESCRIPTION

"Collection of objects used to provide performance information about LMP interfaces and data-bearing links."

::= { lmpGroups 7 }

lmpTeLinkGroup OBJECT-GROUP

```

  OBJECTS { lmpTeLinkNbrRemoteNodeId,
            lmpTeLinkVerification,
            lmpTeLinkFaultManagement,
            lmpTeLinkDwdm,

```

```

        lmpTeLinkOperStatus,
        lmpTeLinkRowStatus,
        lmpTeLinkStorageType,
        lmpTeLinkNotificationsEnabled
    }
    STATUS    current
    DESCRIPTION
        "Objects that can be used to configure TE links."
    ::= { lmpGroups 8 }

lmpDataLinkGroup OBJECT-GROUP
    OBJECTS { lmpDataLinkType,
              lmpDataLinkAddressType,
              lmpDataLinkIpAddr,
              lmpDataLinkRemoteIpAddress,
              lmpDataLinkRemoteIfId,
              lmpDataLinkEncodingType,
              lmpDataLinkActiveOperStatus,
              lmpDataLinkPassiveOperStatus,
              lmpDataLinkRowStatus,
              lmpDataLinkStorageType
            }
    STATUS    current
    DESCRIPTION
        "Collection of objects that represent data-bearing link
        configuration."
    ::= { lmpGroups 9 }

lmpNotificationGroup NOTIFICATION-GROUP
    NOTIFICATIONS { lmpTeLinkPropertyMismatch,
                   lmpDataLinkPropertyMismatch,
                   lmpUnprotected,
                   lmpControlChannelUp,
                   lmpControlChannelDown,
                   lmpTeLinkDegraded,
                   lmpTeLinkNotDegraded,
                   lmpDataLinkVerificationFailure }
    STATUS    current
    DESCRIPTION
        "Set of notifications defined in this module."
    ::= { lmpGroups 10 }

-- End of LMP-MIB
END

```

10. Security Considerations

There are a number of management objects defined in this MIB module 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:

- Unauthorized changes to the `lmpNbrTable`, `lmpControlChannelTable`, `lmpTeLinkTable`, and `lmpDataLinkTable` may disrupt allocation of resources in the network.

Some of the readable objects in this MIB module (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 `lmpNbrTable` exposes the network provider's node IP addresses.
- `lmpControlChannelTable` exposes the network provider's control network.
- `lmpDataLinkTable` exposes the network provider's data network.

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 this MIB module.

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.

11. Contributors

Sudheer Dharanikota

EMail: sudheer@ieee.org

12. Acknowledgements

The general structure of this document has been modeled around the MPLS Label Switching Router (LSR) MIB [RFC3813].

The authors wish to thank Dmitry Ryumkin, Baktha Muralidharan and George Wang.

Thanks to Tom Petch for spotting inconsistencies in RFC 4327 and to Bert Wijnen for document review.

13. IANA Considerations

No new IANA actions are requested in this document. All IANA actions from RFC 4327 still hold and are reproduced here for information.

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

13.1. IANA Considerations for LMP ifType

The IANA has assigned 227 ifType for LMP interfaces.

13.2. IANA Considerations for LMP-MIB

The IANA has assigned { transmission 227 } to the LMP-MIB module specified in this document.

14. Changes from RFC 4327 to RFC 4631

The following changes have been made relative to RFC 4327.

- a. Show that this document obsoletes RFC 4327.
- b. Indicate in Abstract that this document provides minor corrections to RFC 4327.
- c. Correct use of TruthValue settings such that True is always 1, and False is always 2.
- d. Update to acknowledgements section.
- e. Note in IANA section to show no further action required.
- f. Remove identification of RFC 4327 and request RFC Editor to insert new RFC number.
- g. Update timestamps.

- h. Update author information.
- i. Added punctuation to REFERENCE clauses.
- j. Update Revision History clause.
- k. Add this section.
- l. Remove square braces from references to external documents from within the MIB module itself.
- m. Minor editorial corrections to text and DESCRIPTIONS clauses.

15. References

15.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2434] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 2434, October 1998.
- [RFC2578] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.
- [RFC2579] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Textual Conventions for SMIv2", STD 58, RFC 2579, April 1999.
- [RFC2580] McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., and S. Waldbusser, "Conformance Statements for SMIv2", STD 58, RFC 2580, April 1999.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.
- [RFC3471] Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, January 2003.
- [RFC4001] Daniele, M., Haberman, B., Routhier, S., and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 4001, February 2005.
- [RFC4204] Lang, J., "Link Management Protocol (LMP)", RFC 4204, October 2005.

- [RFC4207] Lang, J. and D. Papadimitriou, "Synchronous Optical Network (SONET)/Synchronous Digital Hierarchy (SDH) Encoding for Link Management Protocol (LMP) Test Messages", RFC 4207, October 2005.
- [RFC4209] Fredette, A. and J. Lang, "Link Management Protocol (LMP) for Dense Wavelength Division Multiplexing (DWDM) Optical Line Systems", RFC 4209, October 2005.
- [RFC4220] Dubuc, M., Nadeau, T., and J. Lang, "Traffic Engineering Link Management Information Base", RFC 4220, November 2005.

15.2. Informative References

- [RFC3410] Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework", RFC 3410, December 2002.
- [RFC3813] Srinivasan, C., Viswanathan, A., and T. Nadeau, "Multiprotocol Label Switching (MPLS) Label Switching Router (LSR) Management Information Base (MIB)", RFC 3813, June 2004.

Authors' Addresses

Martin Dubuc

EMail: dubuc.consulting@sympatico.ca

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

EMail: tnadeau@cisco.com

Jonathan P. Lang
Sonos, Inc.
223 E. De La Guerra St.
Santa Barbara, CA 93101

EMail: jplang@ieee.org

Evan McGinnis
Hammerhead Systems
640 Clyde Court
Mountain View, CA 94043

EMail: emcginnis@hammerheadsystems.com

Adrian Farrel
Old Dog Consulting

EMail: adrian@olddog.co.uk

Full Copyright Statement

Copyright (C) The Internet Society (2006).

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 provided by the IETF Administrative Support Activity (IASA).

