IEEE STANDARDS ASSOCIATION

IEEE Standard for
Local and metropolitan area networks—

Station and Media Access Control Connectivity Discovery

IEEE Computer Society

Sponsored by the
LAN/MAN Standards Committee

IEEE
3 Park Avenue
New York, NY 10016-5997
USA

IEEE Std 802.1AB™-2016
(Revision of
IEEE Std 802.1AB-2009)

IEEE Std 802.1AB™-2016
(Revision of
IEEE Std 802.1AB-2009)

IEEE Standard for
Local and metropolitan area networks—

Station and Media Access Control Connectivity Discovery

Sponsor

**LAN/MAN Standards Committee
of the
IEEE Computer Society**

Approved 29 January 2016

IEEE-SA Standards Board

Abstract: A protocol and a set of managed objects that can be used for discovering the physical topology from adjacent stations in IEEE 802® LANs are defined in this document.

Keywords: IEEE 802.1AB™, link layer discovery protocol, management information base, topology discovery, topology information

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

PDF: **ISBN 978-1-5044-0632-1** **STD20761**
Print: **ISBN 978-0-5044-0633-8** **STDPD20761**

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association ("IEEE-SA") Standards Board. IEEE ("the Institute") develops its standards through a consensus development process, approved by the American National Standards Institute ("ANSI"), which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE should be considered the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied on as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board

445 Hoes Lane

Piscataway, NJ 08854 USA

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not imply compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate fee, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every ten years. When a document is more than ten years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: http://standards.ieee.org/findstds/errata/index.html. Users are encouraged to check this URL for errata periodically.

Patents

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE-SA Website at http://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

Participants

At the time this standard was submitted to the IEEE-SA for approval, the IEEE 802.1 Working Group had the following membership:

Glenn Parsons, *Chair*
John Messenger, *Vice Chair and Maintenance Task Group Chair*
Tony Jeffree, *Editor*

Christian Boiger
Paul Bottorff
David Chen
Feng Chen
Weiying Cheng
Rodney Cummings
János Farkas
Norman Finn
Geoffrey Garner
Eric Gray
Craig Gunther
Stephen Haddock
Mark Hantel
Marc Holness
Michael Johas Teener
Hal Keen
Stephan Kehrer
Marcel Kiessling
Philippe Klein
Jouni Korhonen
Yizhou Li
Christophe Mangin
Tom McBeath
James McIntosh
Hiroki Nakano
Bob Noseworthy
Donald R. Pannell
Walter Pienciak
Karen Randall
Maximilian Riegel
Dan Romascanu
Jessy Rouyer
Panagiotis Saltsidis
Michael Seaman
Daniel Sexton
Johannes Specht
Wilfried Steiner
Patricia Thaler
David Thornburg
Jeremy Touve
Paul Unbehagen
Karl Weber
Brian Weis
Jordon Woods
Helge Zinner
Juan Carlos Zuniga

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Thomas Alexander
Butch Anton
Lee Armstrong
Stefan Aust
Christian Boiger
Nancy Bravin
William Byrd
Juan Carreon
Rodney Cummings
Douglas Dorr
János Farkas
Yukihiro Fujimoto
David Gregson
Randall Groves
Craig Gunther
Stephen Haddock
Marek Hajduczenia
Jerome Henry
Marco Hernandez
Guido Hiertz
Werner Hoelzl
Noriyuki Ikeuchi
Sergiu Iordanescu
Atsushi Ito
Raj Jain
Tony Jeffree
Michael Johas Teener
Adri Jovin
Shinkyo Kaku
Piotr Karocki
Stuart Kerry
Yongbum Kim
Paul Lambert
Robert Landman
David Lewis
Arthur H. Light
William Lumpkins
Michael Lynch
Elvis Maculuba
Jonathon McLendon
Richard Mellitz
John Messenger
Charles Moorwood
Michael Newman
Nick S. A. Nikjoo
Satoshi Obara
Alon Regev
Maximilian Riegel
Robert Robinson
Benjamin Rolfe
Dan Romascanu
Jessy Rouyer
John Santhoff
Bartien Sayogo
Michael Seaman
Thomas Starai
Eugene Stoudenmire
Rene Struik
Walter Struppler
Michael Swearingen
Patricia Thaler
Mark-Rene Uchida
Lorenzo Vangelista
Dmitri Varsanofiev
George Vlantis
Khurram Waheed
Karl Weber
Hung-Yu Wei
Natalie Wienckowski
Andreas Wolf
Oren Yuen
Zhen Zhou

When the IEEE-SA Standards Board approved this standard on 29 January 2016, it had the following membership:

John Kulick, *Chair*
Jon Walter Rosdahl, *Vice Chair*
Richard H. Hulett, *Past Chair*
Konstantinos Karachalios, *Secretary*

Masayuki Ariyoshi
Ted Burse
Stephen Dukes
Jean-Phillippe Faure
J. Travis Griffith
Gary Hoffman
Michael Janezic
Joseph L. Koepfinger*
David J. Law
Hung Ling
Andrew Myles
T. W. Olsen
Glenn Parsons
Ronald C. Peterson
Annette D. Reilly
Stephen J. Shellhammer
Adrian P. Stephens
Yatin Trivedi
Phillip Winston
Don Wright
Yu Yuan
Daidi Zhong

*Member Emeritus

Historical participants

Included is a historical list of participants in the IEEE 802.1 Working Group who have dedicated their valuable time, energy, and knowledge to the creation of this material.

The following individuals participated in the 2005 publication of this standard.

Tony Jeffree, *Chair*
Paul Congdon, *Vice Chair*
Bill Lane, *Technical Editor*
Mick Seaman, *Interworking Task Group Chair*

Les Bell
Paul Bottorff
Jim Burns
Marco Carugi
Dirceu Cavendish
Arjan de Heer
Anush Elangovan
Hesham Elbakoury
David Elie-Dit-Cosaque
Norm Finn
David Frattura
Gerard Goubert
Steve Haddock
Ran Ish-Shalom
Atsushi Iwata
Neil Jarvis
Manu Kaycee
Hal Keen
Roger Lapuh
Loren Larsen
Joe Lawrence
Yannick Le Goff
Marcus Leech
Mahalingam Mani
Dinesh Mohan
Bob Moskowitz
Don O'Connor
Don Pannell
Glenn Parsons
Ken Patton
Frank Reichstein
John Roese
Allyn Romanow
Dan Romascanu
Jessy V. Rouyer
Ali Sajassi
Dolors Sala
Muneyoshi Suzuki
Jonathan Thatcher
Michel Thorsen
Dennis Volpano
Karl Weber
Ludwig Winkel
Michael D. Wright

The following individuals participated in the 2009 publication of this standard.

Tony Jeffree, *Chair and Editor*
Paul Congdon, *Vice Chair*
Stephen Haddock, *Chair, Interworking Task Group*

Osama Aboul-Magd
Zehavit Alon
Caitlin Bestler
Jan Bialkowski
Rob Boatright
Jean-Michel Bonnamy
Paul Bottorff
Rudolf Brandner
Craig W. Carlson
Weiying Cheng
Rao Cherukuri
Paul Congdon
Diego Crupnicoff
Claudio Desanti
Zhemin Ding
Linda Dunbar
Hesham M. Elbakoury
David Elie-Dit-Cosaque
János Farkas
Donald Fedyk
Norman Finn
Robert Frazier
John Fuller
Geoffrey Garner
Anoop Ghanwani
Franz Goetz
Yannick Le Goff
Eric Gray
Karanvir Grewal
Craig Gunther
Mitch Gusat
Asif Hazarika
Charles Hudson
Romain Insler
Michael Johas Teener
Abhay Karandikar
Prakash Kashyap
Hal Keen
Keti Kilcrease
Yongbum Kim
Philippe Klein
Mike Ko
Vinod Kumar
Bruce Kwan
Kari Laihonen
Michael Lerer
Gael Mace
Ben Mack-Cran
David Martin
Riccardo Martinotti
Alan McGuire
James McIntosh
Menucher Menuchery
John Messenger
Matthew Mora
Eric Multanen
Kevin Nolish
Hiroshi Ohta
David Olsen
Donald Pannell
Glenn Parsons
Joseph Pelissier
David Peterson
Hayim Porat
Max Pritikin
Ananda Rajagopal
Karen T. Randall
Guenter Roeck
Josef Roese
Derek J. Rohde
Dan Romascanu
Moran Roth
Jessy V. Rouyer
Jonathan Sadler
Ali Sajassi
Joseph Salowey
Panagiotis Saltsidis
Satish Sathe
John Sauer
Michael Seaman
Koichiro Seto
Himanshu Shah
Nurit Sprecher
Kevin B. Stanton
Robert A. Sultan
Muneyoshi Suzuki
George Swallow
Attila Takacs
Patricia Thaler
Oliver Thorp
Manoj Wadekar
Yuehua Wei
Brian Weis
Bert Wijnen
Michael D. Wright
Chien-Hsien Wu
Ken Young
Glen Zorn

The following individuals participated in the development of Corrigendum 1 to the 2009 publication of this standard.

Tony Jeffree, *Chair and Editor*
Glenn Parsons, *Vice-Chair and Maintenance Task Group Chair*

Ting Ao
Kenneth Boehlke
Christian Boiger
Brad Booth
Paul Bottorff
Jeffrey Catlin
Xin Chang
Weiying Cheng
Diego Crupnicoff
Rodney Cummings
Donald Eastlake, III
János Farkas
Donald Fedyk
Norman Finn
Andre Fredette
Geoffrey Garner
Anoop Ghanwani
Franz Goetz
Mark Gravel
Eric Gray
Robert Grow
Yingjie Gu
Craig Gunther
Stephen Haddock
Hitoshi Hayakawa
Mirko Jakovljevic
Markus Jochim
Michael Johas Teener
Girault Jones
Daya Kamath
Hal Keen
Yongbum Kim
Philippe Klein
Oliver Kleineberg
Jeff Lynch
Ben Mack-Crane
John Messenger
Eric Multanen
Henry Muyshondt
David Olsen
Donald Pannell
Karen Randall
Josef Roese
Dan Romascanu
Jessy Rouyer
Ali Sajassi
Panagiotis Saltsidis
Rick Schell
Michael Seaman
Koichiro Seto
Daniel Sexton
Rakesh Sharma
Johannes Specht
Kevin Stanton
Wilfried Steiner
Patricia Thaler
Jeremy Touve
Albert Tretter
Maarten Vissers
Yuehua Wei
Min Xiao

The following individuals participated in the development of Corrigendum 2 to the 2009 publication of this standard.

Glenn Parsons, *Working Group Chair*
John Messenger, *Vice-Chair and Maintenance Task Group Chair*
Tony Jeffree, *Editor*

Ting Ao
Christian Boiger
Paul Bottorff
David Chen
Feng Chen
Weiying Cheng
Diego Crupnicoff
Rodney Cummings
Patrick Diamond
Aboubacar Kader Diarra
János Farkas
Norman Finn
Geoffrey Garner
Anoop Ghanwani
Mark Gravel
Eric W. Gray
Craig Gunther
Stephen Haddock
Hitoshi Hayakawa
Jeremy Hitt
Rahil Hussain
Michael Johas Teener
Peter Jones
Hal Keen
Marcel Kiessling
Yongbum Kim
Philippe Klein
Jouni Korhonen
Jeff Lynch
Ben Mack-Crane
Christophe Mangin
James McIntosh
Eric Multanen
Donald Pannell
Karen Randall
Maximilian Riegel
Dan Romascanu
Jessy V. Rouyer
Panagiotis Saltsidis
Behcet Sarikaya
Michael Seaman
Daniel Sexton
Johannes Specht
Kevin B. Stanton
Wilfried Steiner
Vahid Tabatabaee
Patricia Thaler
Jeremy Touve
Karl Weber
Yuehua Wei
Brian Weis
Jordon Woods
Juan-Carlos Zuniga

Introduction

This introduction is not part of IEEE Std 802.1AB™-2016, IEEE Standard for Local and metropolitan area networks—Station and Media Access Control Connectivity Discovery.

This revision of IEEE Std 802.1AB does not include any new functionality. It simply incorporates the following into the base text of the 2009 revision:

— IEEE Std 802.1AB-2009/Cor 1-2013
— IEEE Std 802.1AB-2009/Cor 2-2015

Three annexes from the 2009 revision have been deleted:

— Annex D (Using LLDP to detect potential communication problems) was deleted because it was considered to be no longer useful.
— Annex E and Annex F were deleted because the material in them can be found in Annex D of IEEE Std 802.1Q-2014 and Clause 79 of IEEE Std 802.3-2012, respectively.

The bibliography (found in Annex G of the 2009 revision) is Annex D in this revision.

Contents

List of figures

List of tables

IEEE Standard for
Local and metropolitan area networks—

Station and Media Access Control Connectivity Discovery

1. Overview

The Link Layer Discovery Protocol (LLDP) specified in this standard allows stations attached to an IEEE 802® LAN to advertise, to other stations attached to the same IEEE 802 LAN, the major capabilities provided by the system incorporating that station, the management address or addresses of the entity or entities that provide management of those capabilities, and the identification of the station's point of attachment to the IEEE 802 LAN required by those management entity or entities.

The information distributed via this protocol is stored by its recipients in a standard Management Information Base (MIB), making it possible for the information to be accessed by a Network Management System (NMS) using a management protocol such as the Simple Network Management Protocol (SNMP).

1.1 Scope

The scope of this standard is to define a protocol and management elements, suitable for advertising information to stations attached to the same IEEE 802 LAN, for the purpose of populating physical topology and device discovery management information databases. The protocol facilitates the identification of stations connected by IEEE 802 LANs/MANs, their points of interconnection, and access points for management protocols.

This standard defines a protocol that

a) Advertises connectivity and management information about the local station to adjacent stations on the same IEEE 802 LAN.
b) Receives network management information from adjacent stations on the same IEEE 802 LAN.
c) Operates with all IEEE 802 access protocols and network media.
d) Establishes a network management information schema and object definitions that are suitable for storing connection information about adjacent stations.
e) Provides compatibility with the IETF PTOPO MIB (IETF RFC 2922 [B9]).[1]

1.2 Purpose

An IETF MIB (IETF RFC 2922 [B9]) and a number of vendor specific MIBs have been created to describe a network's physical topology and associated systems within that topology.

This standard specifies the necessary protocol and management elements to

a) Facilitate multi-vendor inter-operability and the use of standard management tools to discover and make available physical topology information for network management.
b) Make it possible for network management to discover certain configuration inconsistencies or malfunctions that can result in impaired communication at higher layers.
c) Provide information to assist network management in making resource changes and/or re-configurations that correct configuration inconsistencies or malfunctions identified in b) above.

[1]The numbers in brackets correspond to those in the bibliography in Annex D.

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in the text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

IEEE Std 802®, IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture.[2, 3]

IEEE Std 802.1AC™, IEEE Standard for Local and metropolitan area networks—Media Access Control (MAC) Service Definition.

IEEE Std 802.1AE™, IEEE Standard for Local and Metropolitan Area Networks—Media Access Control (MAC) Security.

IEEE Std 802.1AX™, IEEE Standard for Local and Metropolitan Area Networks—Link Aggregation.

IEEE Std 802.1Q™, IEEE Standards for Local and Metropolitan Area Networks: Bridges and Bridged Networks.

IEEE Std 802.1X™, IEEE Standard for Local and Metropolitan Area Networks—Port-Based Network Access Control.

IEEE Std 802.3™, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.

IETF RFC 2863, The Interfaces Group MIB, June 2000.[4]

IETF RFC 3046, DHCP Relay Agent Information Option, January 2001.

IETF RFC 3232, Assigned Numbers: RFC 1700 is Replaced by an On-line Database, January 2002.[5]

IETF RFC 3410, Introduction and Applicability Statements for Internet Standard Management Framework, December 2002.

IETF RFC 3417, Transport Mappings for the Simple Network Management Protocol (SNMP), December 2002.

IETF RFC 3418, Management Information Base (MIB) for the Simple Network Management Protocol (SNMP), December 2002.

IETF RFC 3629, UTF-8, a transformation format of ISO 10646, November 2003.

IETF RFC 4502, Remote Network Monitoring Management Information Base Version 2, May 2006.

---

[2]IEEE and 802 are registered trademarks in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

[3]IEEE publications are available from The Institute of Electrical and Electronics Engineers (http://standards.ieee.org).

[4]IETF documents (i.e., RFCs) are available for download at http://www.rfc-archive.org/.

[5]The IETF RFC 3232 ianaAddressFamilyNumbers on-line database module is accessible at http://www.iana.org.

IETF RFC 4639, Cable Device Management Information Base for Data-Over-Cable Service Interface Specification (DOCSIS) Compliant Cable Modems and Cable Modem Termination Systems, December 2006.

IETF RFC 4789, Simple Network Management Protocol (SNMP) over IEEE 802 Networks, November 2006.

IETF RFC 6933, Entity MIB (Version 4), May 2013.

ISO/IEC 8824-1 [ITU-T Rec. X.680 (2002)], Abstract Syntax Notation One (ASN.1): Specification of Basic Notation.[6]

[6]ASN.1 standards are available at http://asn1.elibel.tm.fr/en/standards/index.htm#asn1.

3. Definitions and numerical representation

3.1 Definitions

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.[7]

alpha-numeric information: Information that is encoded using the 8-bit Universal Character Set (UCS)/ Unicode Transformation Format (UTF-8) octet sequence [IETF RFC 3629].

chassis: A physical component incorporating one or more IEEE 802® LAN stations and their associated application functionality.

chassis identifier: An administratively assigned name that identifies the particular chassis within the context of an administrative domain that comprises one or more networks.

IEEE 802® LAN: Local area network (LAN) technologies that provide a media access control (MAC) Service equivalent to the MAC Service defined in IEEE Std 802.1AC.

IEEE 802® LAN station: An IEEE 802-compatible entity that incorporates all the necessary mechanisms to participate in media access control of an IEEE 802 LAN, and that is at least capable of providing the MAC service plus the mandatory capabilities of the LLC.

NOTE 1—For example, routers and host computers are stations in a bridged network. Bridges, in addition to their relay function, also include station capabilities.[8]

Link Layer Discovery Protocol (LLDP): A media-independent protocol capable of running on all IEEE 802® LAN stations and to allow an LLDP agent to learn the connectivity and management information from adjacent stations.

LLDP agent: The protocol entity that implements LLDP for a particular MSAP associated with a Port.

MAC service access point (MSAP): The access point for MAC services provided to the LLC sublayer.

MSAP identifier: The identifier of a MAC service access point.

NOTE 2—In this standard, the concatenation of the chassis identifier and the port identifier is used by LLDP as an MSAP identifier, to identify the port associated with an IEEE 802® LAN station.

management entity: The protocol entity that implements a particular network management protocol and that provides access support to a MIB associated with the protocol and implemented on a host chassis.

Management Information Base (MIB): The instantiation of all MIB modules in a managed entity (e.g., system or device).

Management Information Base module (MIB module): The specification or schema for a data base that can be populated with the information required to support a network management information system.

network: An interconnected group of systems, each comprising one or more IEEE 802® LAN stations.

---

[7] *IEEE Standards Dictionary Online* subscription is available at: http://www.ieee.org/portal/innovate/products/standard/standards_dictionary.html.

[8] Notes in text, tables, and figures are given for information only and do not contain requirements needed to implement the standard.

Network Management System (NMS): A management system that is capable of utilizing the information in a MIB.

object identifier (OID): An identifier used to name an object. Structurally, an OID consists of a node in a hierarchically-assigned namespace, formally defined in ISO/IEC 8824-1, Abstract Syntax Notation 1 (ASN.1). OIDs are used in this standard to identify MIB modules and the objects they contain.

physical network topology: The identification of systems, of IEEE 802® LAN stations that compose each system, and of the IEEE 802 LAN stations that attach to the same IEEE 802 LAN.

port: The entity in a chassis/system to support an MSAP. A port incorporates one and only one MSAP and identifies the collection of manageable entities that provide the MAC Service at the MSAP.

port identifier: An administratively assigned name that identifies the particular port within the context of a system, where the identification is convenient, local to the system, and persistent for the system's use and management (whereas the MAC address that globally identifies the MSAP can not be).

system: A managed collection of hardware and software components incorporating one or more chassis, stations and ports.

station only: A non-forwarding IEEE 802® LAN station such as a user workstation, network file server, or print server.

type, length, value (TLV): A short, variable length encoding of an information element consisting of sequential type, length, and value fields where the type field identifies the type of information, the length field indicates the length of the information field in octets, and the value field contains the information, itself.

3.2 Numerical representation

Decimal, hexadecimal, and binary numbers are used within this document. For clarity, decimal numbers are generally used to represent counts, hexadecimal numbers are used to represent addresses, and binary numbers are used to describe bit patterns within binary fields.

Decimal numbers are represented in their usual 0, 1, 2, … format. Hexadecimal numbers are represented by a string of one or more hexadecimal (0–9, A–F) digits followed by the subscript 16, except in C-code contexts, where they are written as `0x123EF2`, etc. Binary numbers are represented by a string of one or more binary (0,1) digits, followed by the subscript 2. Thus the decimal number "26" may also be represented as "$1A_{16}$" or "11010_2".

MAC addresses, EtherType values, and OUI/EUI values are represented as strings of 8-bit hexadecimal numbers separated by hyphens and without a subscript, as, for example, "01-80-C2-00-00-15" or "AA-55-11", using the hexadecimal representation defined in IEEE Std 802.

4. Acronyms and abbreviations

AP	Access Point
BSSID	basic service set identification
DA	Destination Address
DS	Distribution System
EPD	Ethertype Protocol Discrimination
EUI	Extended Unique Identifier
FCS	Frame Check Sequence
IANA	Internet Assigned Numbers Authority
ID	Identifier
IETF	Internet Engineering Task Force
KaY	Media access control Security Key Agreement Entity
LLC	logical link control (sublayer)
LLDP	Link Layer Discovery Protocol
LLDPDU	Link Layer Discovery Protocol data unit
LSAP	link service access point
MAC	media access control (sublayer)
MAU	medium attachment unit
MDI	media dependent interface
MIB	Management Information Base (module)
MSAP	Media access control service access point
MSDU	Media access control service data unit
NMS	Network Management System
OID	object identifier
OUI	organizationally unique identifier
PD	Powered Device
PHY	physical (sublayer)

PMD	physical media dependent (sublayer)
PSE	Power Sourcing Equipment
PVID	port virtual local area network identifier
PPVID	port and protocol virtual local area network identifier
PTOPO	the name of the Internet Engineering Task Force physical topology Management Information Base
RFC	Request for comments
SA	Source Address
SecY	Media access control Security Entity
SNAP	Subnetwork Access Protocol
SNMP	Simple Network Management Protocol
STP	Spanning Tree Protocol
TPMR	Two-port Media access control relay
TLV	type, length, value
TTL	time to live (value)
UCS	Universal Character Set
UTF-8	8-bit Universal Character Set/Unicode Transformation Format
VID	virtual local area network identifier

5. Conformance

This clause specifies the mandatory and optional capabilities provided by conformant implementations of this standard.

5.1 Terminology

For consistency with IEEE and existing IEEE 802.1™ standards terminology, requirements placed upon conformant implementations of this standard are expressed using the following terminology:

a) *Shall* is used for mandatory requirements.
b) *May* is used to describe implementation or administrative choices ("may" means "is permitted to," and hence, "may" and "may not" mean precisely the same thing).
c) *Should* is used for recommended choices (the behaviors described by "should" and "should not" are both permissible but not equally desirable choices).

The PICS proforma (see Annex A) reflects the occurrences of the words *shall*, *may*, and *should* within the standard.

The standard avoids needless repetition and apparent duplication of its formal requirements by using is, is not, are, and are not for definitions and the logical consequences of conformant behavior. Behavior that is permitted but is neither always required nor directly controlled by an implementor or administrator, or whose conformance requirement is detailed elsewhere, is described by can. Behavior that never occurs in a conformant implementation or system of conformant implementations is described by can not. The word *allow* is used as a replacement for the phrase "support the ability for," and the word *capability* means "is able to, or can be configured to."

5.2 Protocol Implementation Conformance Statement (PICS)

The supplier of an implementation that is claimed to conform to this standard shall complete a copy of the PICS proforma provided in Annex A and shall provide the information necessary to identify both the supplier and the implementation.

5.3 Required capabilities

A system for which conformance to this standard is claimed shall, for all ports for which support is claimed, include the following capabilities:

a) If port access is controlled by IEEE Std 802.1X,[9] LLDP exchanges shall be supported through the controlled port, as specified in Clause 6.
b) The destination and source addressing shall conform to 7.1 and 7.2.
c) EtherType encapsulation shall conform to 7.3.
d) LLDPDU recognition and reception shall conform to the operation of the receive state machine as defined in 9.2.9.
e) LLDPDU encapsulation shall conform to the specifications in 8.2.
f) The basic TLV format capability shall be implemented as defined in 8.4.
g) The basic management set of TLVs shall be implemented as defined in 8.5.
h) The Organizationally Specific TLV format capability shall be implemented as defined in 8.6.

[9]Information on references can be found in Clause 2.

i) The protocol shall conform to the specifications for all Clause 9 subclauses indicated in Table 9-1 for the particular operating mode (transmit only, receive only, or transmit and receive) being implemented.
j) If receipt of LLDPDUs is supported, for every set of TLVs (the basic management set and any organizationally specific sets) supported, support shall be implemented for receipt of every TLV defined in the set.
k) If transmission of LLDPDUs is supported, for every set of TLVs (the basic management set and any organizationally specific sets) supported, support shall be implemented for transmission of every TLV defined in the set.
l) If transmission of LLDPDUs is supported, for every set of TLVs (the basic management set and any organizationally specific sets) supported, a capability shall be implemented for users to determine which optional TLVs are transmitted in any particular LLDPDU.
m) If SNMP is supported, then
    1) The system shall conform to the LLDP management specifications in Clause 11 and shall implement the sections of version 2 of the basic LLDP MIB indicated in Table 11-1 for the operating mode being implemented.
    2) The system shall support at least one of the transport mappings defined by IETF RFC 3417 or IETF RFC 4789.
n) If SNMP is not supported, the system shall provide storage and retrieval capability equivalent to the functionality specified in 10.1 for the operating mode being implemented.

5.4 Optional capabilities

A system for which conformance to this standard is claimed may, for each port for which support is claimed, support the following capabilities:

a) If port access is controlled by IEEE Std 802.1X, LLDP exchanges may be supported through the uncontrolled port, as specified in Clause 6.

6. Principles of operation

LLDP is a link layer protocol that allows an IEEE 802 LAN station to advertise the capabilities and current status of the system associated with an MSAP. The MSAP provides the MAC service to an LLC Entity, and that LLC Entity provides an LSAP to an LLDP agent that transmits and receives information to and from the LLDP agents of other stations attached to the same LAN. The information distributed and received in each LLDPDU is stored in one or more Management Information Bases (MIBs). Figure 6-1 illustrates the LLDP agent and its relationship to its LLC Entity and MSAP, and to additional MIBs designed by the IETF, IEEE 802, and others.

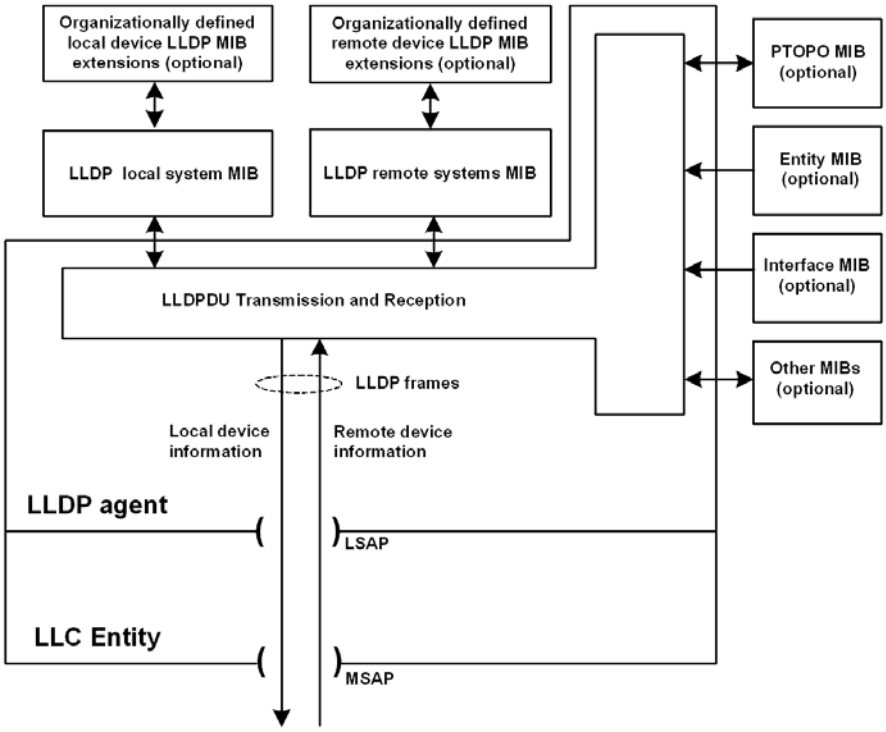

Figure 6-1—LLDP agent and its relationship to its LLC entity

This clause describes general principles of LLDP operation. The following clauses specify transmission, reception, and addressing of LLPDUs (Clause 7); LLDPDU formats (Clause 8); operation of each LLDP agent in detail including state machines (Clause 9); management of LLDP (Clause 10); and the MIB module used for LLDP management (Clause 11).

NOTE 1—For the purposes of this standard, the terms “LLC” and “LLC entity” include the service provided by the operation of entities that support protocol discrimination using an EtherType; i.e., protocol discrimination based on the Type interpretation of the Length/Type field as specified in IEEE Std 802.3, or the use of the SNAP encapsulation of EtherTypes as described in IEEE Std 802. Hence, “LSAP” in the above description can refer to the use of IEEE Std 802.2 LLC addresses, or an EtherType, as a means of higher layer protocol identification.

NOTE 2—The LLC Entity also provides distinct LSAPs to other higher layer protocol entities (such as those responsible for the Spanning Tree Protocol and IP). In a bridge, the LLC Entity associated with each Bridge Port is modeled as being directly connected to the attached LAN, as discussed in 8.13.9 of IEEE Std 802.1Q-2014, so the operation of LLDP is unaffected by the spanning tree Port State.

6.1 Transmission and reception

The information fields in each LLDP frame are contained in a Link Layer Discovery Protocol Data Unit (LLDPDU) as a sequence of variable length information elements, that each include type, length, and value fields (known as TLVs), where

a) Type identifies what kind of information is being sent.
b) Length indicates the length of the information string in octets.
c) Value is the actual information that needs to be sent (for example, a binary bit map or an alpha-numeric string that can contain one or more fields).

Each LLDPDU contains the following three mandatory TLVs (see Table 8-1), and can contain optional TLVs as selected by network management:

d) A Chassis ID TLV.
e) A Port ID TLV.
f) A Time To Live TLV.
g) Zero or more optional TLVs, as allowed by the maximum size of the LLDPDU.
h) An optional End Of LLDPDU TLV.

The chassis ID and the port ID values are concatenated to form a logical MSAP identifier that is used by the recipient to identify the sending LLDP agent/port. Both the chassis ID and port ID values can be defined in a number of convenient forms. Once selected, however, the chassis ID/port ID value combination remains the same as long as the particular port remains operable.

NOTE 1—The statement above is true for any LLDP agent; however, there can be multiple LLDP agents sending and receiving LLDPDUs using different MAC addresses.

A non-zero value in the time to live (TTL) field of the Time To Live TLV tells the receiving LLDP agent how long all information pertaining to this LLDPDU's MSAP identifier is valid so that all the associated information can later be automatically discarded by the receiving LLDP agent if the sender fails to update it in a timely manner. A zero value indicates that any information pertaining to this LLDPDU's MSAP identifier is to be discarded immediately.

NOTE 2—A TTL value of zero can be used, for example, to signal that the sending port has initiated a port shutdown procedure.

The End Of LLDPDU TLV marks the end of the LLDPDU.

The format for the LLDPDU is defined in 8.2. The TLV categories and the basic TLV format are defined in 8.3 and 8.4. The specific format and field contents for the Chassis ID TLV are defined in 8.5.2; for the Port ID TLV, in 8.5.3; for the Time To Live TLV in 8.5.4; and for the End Of LLDPDU TLV, in 8.5.1.

6.2 LLDP operational modes

LLDP is a one way protocol. An LLDP agent can transmit information about the capabilities and current status of the system associated with its MSAP identifier. The LLDP agent can also receive information about the capabilities and current status of the system associated with a remote MSAP identifier. LLDP does not itself contain a mechanism for soliciting specific information from other LLDP agents, nor does it provide a specific means of confirming the receipt of information.

NOTE—The LLDP protocol is designed to advertise information useful for discovering pertinent information about a remote port and to populate topology MIBs. It is not intended to act as a configuration protocol for remote systems, nor as a mechanism to signal control information between ports. During the operation of LLDP, it may be possible to

discover configuration inconsistencies between systems on the same IEEE 802 LAN. LLDP does not provide a mechanism to resolve those inconsistencies. Rather, it provides a means to report discovered information to higher layer management entities.

LLDP allows the transmitter and the receiver to be separately enabled, making it possible to configure an implementation so the local LLDP agent can either transmit only or receive only, or so the local LLDP agent can transmit and receive LLDP information.

6.3 LLDP information categories

The following basic management TLV set (this set is required in all LLDP implementations) is defined by this standard and can be used to describe the system and/or to assist in the detection of configuration inconsistencies associated with the MSAP identifier:

a) Port Description TLV.
b) System Name TLV.
c) System Description TLV.
d) System Capabilities TLV (indicates both the system's capabilities and its current primary network function, such as end station, bridge, router).
e) Management Address TLV.

Table 8-1 includes a list of the optional TLVs in the basic management set and provides subclause references for their specific definitions.

Organizationally Specific TLVs can be defined by either the professional organizations or the individual vendors that are involved with the particular functionality being implemented within a system. The basic format and procedures for defining Organizationally Specific TLVs are provided in 8.6.

6.4 TLV selection

Information for constructing the various TLVs to be sent is stored in the LLDP local system MIB. The selection of which particular TLVs to send is under control of network management. Information received from remote LLDP agents is stored in the LLDP remote systems MIB.

6.5 Transmission principles

Transmission can be initiated either by the expiration of a transmit countdown timing counter or by a change in the value of one or more of the information elements (managed objects) associated with the local system. When a transmit cycle is initiated, the LLDP management entity extracts the managed objects from the LLDP local system MIB and formats this information into TLVs. The TLVs are inserted into an LLDPDU that is passed to the LLDP transmit module. The LLDP transmit module prepends addressing parameters to the LLDPDU as defined in 8.2, 8.3, and 8.4. The LLDPDU and TLV formats are defined in Clause 8. The LLDP transmit state machine is described in 9.2.1 and 9.2.8.

NOTE 1—Because a transmission cycle can be initiated whenever a change occurs within the LLDP local system MIB, it is possible that a series of successive changes over a short period of time could trigger a number of LLDP frames to be sent, each reporting only a single change. LLDP utilizes a transmission delay timer that can be set by network management to ensure that there is a defined maximum number of LLDPDU transmissions in a given time period.

NOTE 2—Under normal circumstances, the information in the receiving LLDP agent's remote systems MIB is refreshed periodically to avoid being discarded due to ageing. To prevent the receiving LLDP agent's remote systems MIB information being aged out because a refresh frame has been lost in transmission, the sending LLDP agent typically sets the TTL value so that several refresh cycles can occur before the received MIB information ages out.

LLDP can disclose information about a device and network that could be considered sensitive in certain environments. Implementers are encouraged to provide controls that take into account the link protection characteristics when including information in an LLDP message. Different information can be included if an LLDP message is being sent on the uncontrolled port, the controlled port, or a MACsec protected connectivity association.

6.6 Reception principles

The LLDP receive module uses the services of the LLC entity to recognize that the incoming MA_UNITDATA.indication contains the correct combination of destination address and MSDU header values to identify it as resulting from a received LLDPDU. The LLDPDU recognition procedures are defined in 7.4 and 9.2.7.7.

6.6.1 LLDPDU and TLV error handling

The LLDPDU is checked to ensure that it contains the correct sequence of three mandatory TLVs at the beginning of the frame (Chassis ID TLV, Port ID TLV, and Time To Live TLV) and then each optional TLV is validated in succession. LLDPDUs that contain detectable errors in the first three mandatory TLVs are discarded. Optional TLVs that contain detectable errors are discarded [see 9.2.7.7.2 c)]. TLVs that are not recognized, but that also contain no basic format errors, are assumed to be valid and are stored for possible later retrieval by network management (see 9.2.7.7.1 and 9.2.7.4). If the End Of LLDPDU TLV is present, any octets that follow it are discarded.

TLVs in which the information string length field contains a value that is greater than the sum of the lengths of the fields within the information string are not discarded.

NOTE—This approach allows later versions of a TLV to define additional fields at the end of the information string; implementations based on an earlier version of the TLV can therefore continue to process the fields that were defined in that version and ignore any new fields added at the end of the information string in later versions. Any information contained in such new fields is not made available to network management protocols via the standard MIB.

6.6.2 LLDP remote systems MIB update

The LLDP remote systems MIB is updated after all TLVs have been validated. LLDP remote system MIB update procedures are defined in 9.2.7.7.4, 9.2.7.7.5, and 9.2.7.4. The LLDP receive state machine is described in 9.2.1 and 9.2.9.

6.7 Systems with multiple LLDP Agents

Each LLDP agent advertises a single set of information in the various TLVs it encodes in each transmitted LLPDU, and is associated with the MSAP that supports the LLC entity that the agent uses to transmit and receive. Each LLDP agent uses its LSAP directly, without the use of any additional multiplexing or addressing above the LSAP to support the use of that LSAP by multiple agents; and each LLC entity provides service to one and only one protocol entity at each of its LSAPs that it supports, using the service provided by a single MSAP. It follows that each LLDP agent makes use of a unique MSAP, and that the agent can be uniquely identified by the receiving agent using the MSAP's identifier as specified in 6.2. A single LLDP management entity can support the operation of multiple LLDP agents within the same system.

Figure 6-2 illustrates the relationship between the LLDP agents, LLC Entities, MSAPs, and the LLDP management entity.

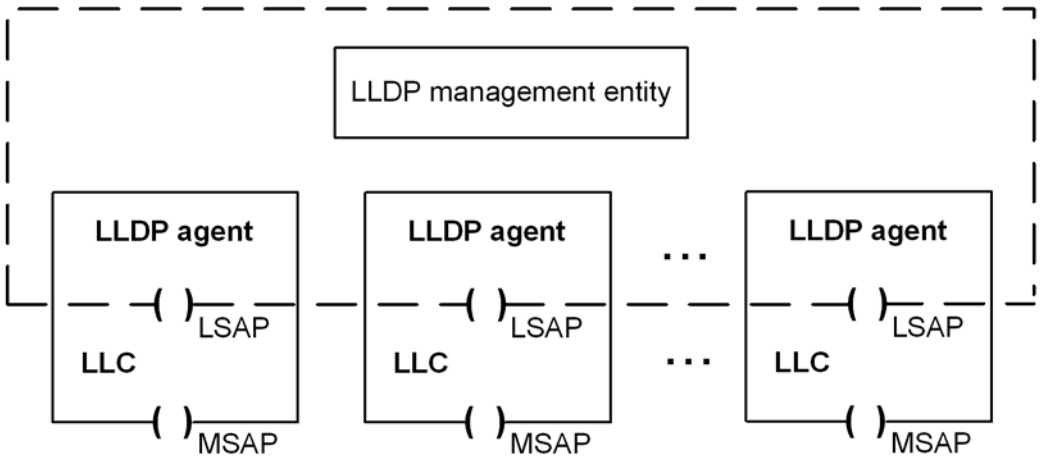

Figure 6-2—Relationship between LLDP agents, LLC Entities, MSAPs, and the LLDP management entity

A given system can require the use of multiple LLDP agents because it naturally comprises multiple MSAPs, or because it needs to be able to transmit different information (different sets of TLVs) to different sets of peers. These sets of peers can be determined by the way that the MAC service is supported within the system (see IEEE Std 802.1AC-2015 for a description of the connectionless connectivity that characterizes an IEEE 802 LAN). A MAC Bridge (see IEEE Std 802.1Q), for example, attaches to multiple LANs, each providing the MAC service at a distinct MSAP. A different example is a station that uses port-based network access control (IEEE Std 802.1X) to provide both a Controlled Port and an Uncontrolled Port, i.e., two distinct MSAPs with potentially different connectivity, for transmission and reception to and from a single LAN. Similarly, different destination addresses can be associated with different sets of potential recipients. Table 7-1 identifies group MAC addresses that can be used for LLDPDU transmission, and 7.1 describes the different transmission scopes associated with each address. LLDP can also be used in conjunction with individual MAC addresses, and with other group MAC addresses. Distinct LLDP agents, and hence separate and distinct protocol state machines, local and remote MIB tables, and MSAPs are maintained for each LLDPDU destination address, even if the use of two or more MSAPs can result in transmission and reception on the same LAN.

NOTE—For a given MAC address that is used as a destination address in received LLDPDUs, there is the possibility for a station to receive LLDPDUs from multiple senders. All LLDPDUs that carry that destination MAC address are received by a single LLDP agent, via a single MSAP; however, as the LLPDU carries information that identifies the source of the LLDPDU, information carried in LLPDUs from different senders is able to be recorded independently in the remote systems MIB.

Figure 6-3 illustrates the use of LLDP in a MAC Bridge, with an LLDP agent for each of the two ports shown. Figure 6-4 shows the use of LLDP in an end system with port-based network access control (IEEE Std 802.1X) supported by MACsec (IEEE Std 802.1AE); an LLDP agent is supported by the Controlled Port and an additional LLDP agent may be supported by the Uncontrolled Port. Figure 6-5 shows LLDP in a MAC Bridge that uses port-based network access control on both ports.

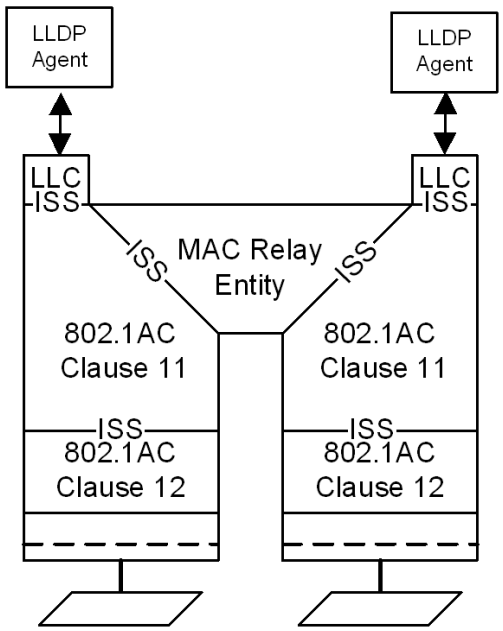

Figure 6-3—LLDP in a MAC Bridge

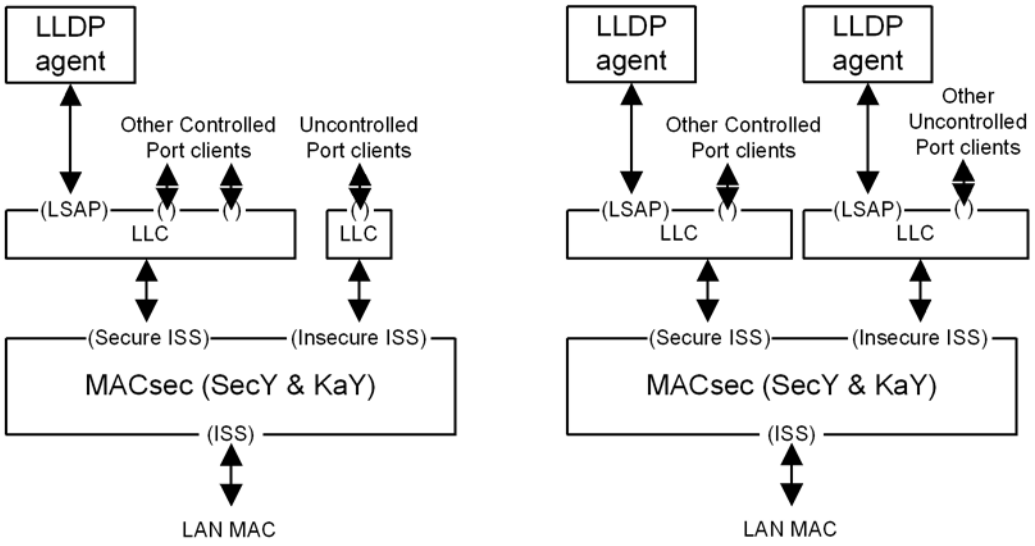

Figure 6-4—LLDP in an end system with port-based network access control

LLDP frames sent on an unprotected link may be modified by an attacker. If an implementation is going to take action based upon a received LLDP attribute, it is advisable to take into account whether the message was received on a protected link, and allow for the system to be configured such that only information received in protected messages is acted upon.

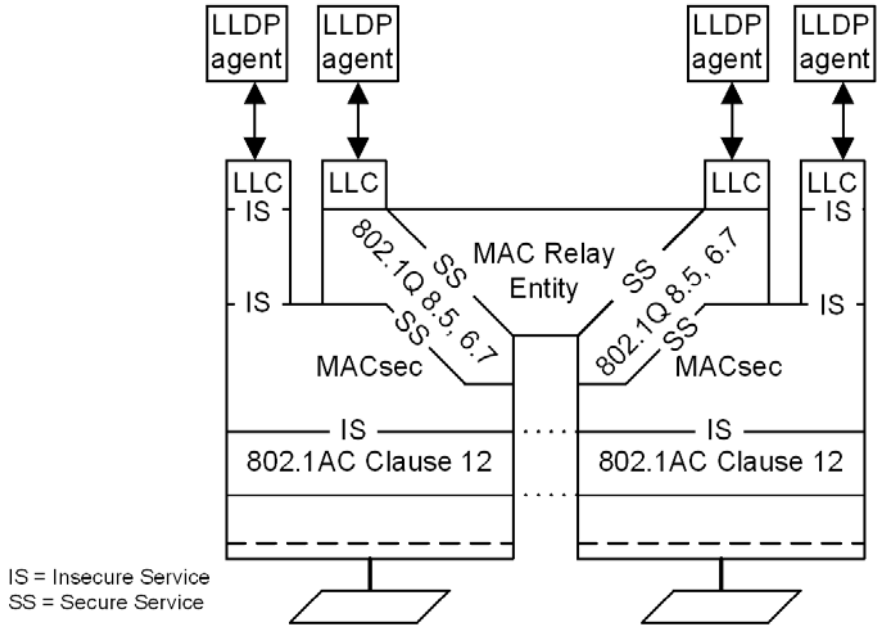

Figure 6-5—LLDP in a MAC Bridge that uses port-based network access control on both ports

The group addresses identified in Table 7-1 are taken from the set specified as reserved addresses by bridging standards. Frames (including those conveying LLDPDUs) that use these destination addresses are filtered or not by the various types of bridge components, as specified by Table 8-1 through Table 8-3 of IEEE Std 802.1Q-2014. The choice of a particular address thus allows a given agent to limit the propagation of LLDPDUs to an individual LAN, or to allow them a wider scope. Figure 6-6 illustrates the various scopes achievable with the three destination group MAC addresses identified in Table 7-1, their use is specified further in 7.1.

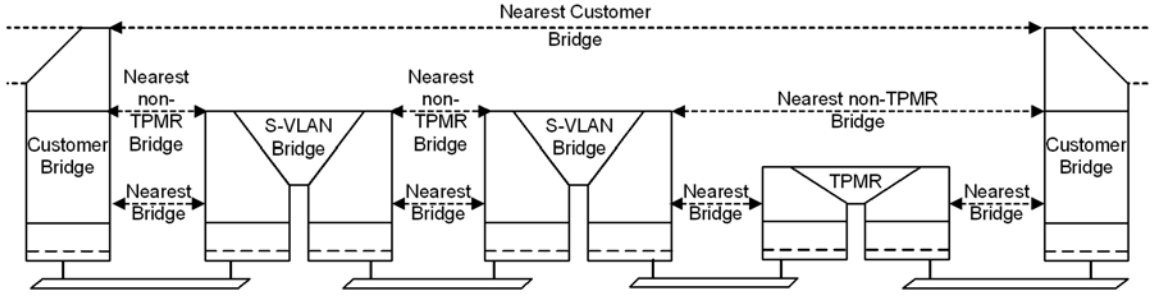

Figure 6-6—Scope of group MAC addresses

More than one LLDP agent, each using a different address scope, can be instantiated for a given system port by adding a simple shim that provides the necessary distinct MSAPs by multiplexing and demultiplexing between those MSAPs and a common MSAP for the port, as illustrated in Figure 6-7.

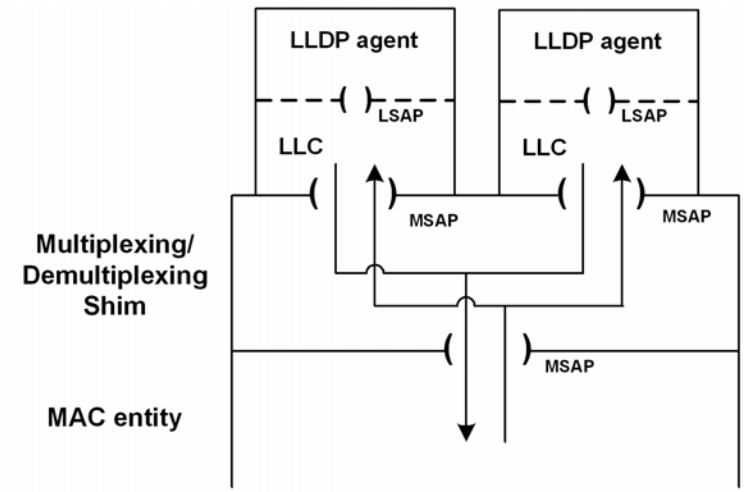

Figure 6-7—Multiplexing and demultiplexing using shims

6.8 LLDP and Link Aggregation

An LLDP agent can be configured on an IEEE 802.1AX™ Aggregated Port and/or on any number of the physical ports belonging to the Aggregation. Some TLVs (e.g., the Power Via MDA TLV defined in 79.3.2 of IEEE Std 802.3-2012) only make sense when transmitted from an LLDP agent configured on a physical port, and can cause incorrect operation if configured on an Aggregation. Other TLVs (e.g., the DCBX TLVs defined in 38.4 of IEEE Std 802.1Q-2014) make sense only when transmitted from an Aggregated Port, because they carry information applicable to the simulated single interface represented by the Aggregation. Not all standards that define LLDP TLVs are careful to state whether each TLV is applicable to an Aggregation, to a physical port, or to both. Only LLDP TLVs that are appropriate for transmission on a given Port should be transmitted on that Port.

An LLDP agent configured on an Aggregated Port, or on one of the physical ports of an Aggregation, transmits the Link Aggregation TLV, and processes received LLDPDUs, as specified in F.1 of IEEE Std 802.1AX-2014. An LLDP agent configured on a physical port of an Aggregation uses the LLDP Parser/Multiplexer as specified in 6.1.3 of IEEE Std 802.1AX-2014 for the transmission and reception of LLDPDUs containing TLVs that are specific to that physical port. Since a port not running Link Aggregation is equivalent to an Aggregation with a single physical port, any LLDP agent can transmit the Link Aggregation TLV and process received LLDPDUs as specified in 802.1AX.

NOTE—The consideration of aggregated vs. Physical ports is similar to the distinction among destination MAC addresses discussed in 7.1. In both cases, LLDP TLVs that carry information specific to a physical link should be used only in the LLDP agent with the shortest reach.

7. LLDPDU transmission, reception, and addressing

This standard is intended to be compatible with all IEEE 802 MACs.

LLDP uses the service provided by the LLDP/LSAP and LLC to transmit and receive LLDPDUs. Each LLDPDU is transmitted as a single MAC service request by an LLC entity that uses a single instance of the MAC Service provided at an MSAP. Each incoming LLDP frame is received at the MSAP by the LLC entity as a MAC service indication.

NOTE—For the purposes of this standard, the terms “LLC” and “LLC entity” include the service provided by the operation of entities that support protocol discrimination using an EtherType, i.e., Ethertype Protocol Discrimination (EPD) as specified in IEEE Std 802.

The parameters of each service request and service indication comprise

a) Destination address
b) Source address
c) EtherType
d) LLDPDU

The LLDP EtherType, used to identify the LLDP protocol, is prepended to the LLDPDU as shown in Figure 7-1 to form the MSDU of the corresponding MAC service request.

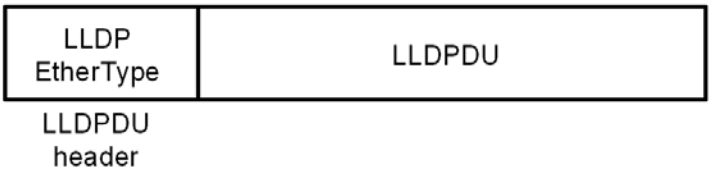

Figure 7-1—MSDU format

The values of the parameters used by LLDP, and their encoding by the LLC entity that supports the LLDP LSAP, are specified in the following subclauses.

7.1 Destination address

A set of standard group MAC addresses that can be used by LLDP is specified in Table 7-1. These addresses are in the range of IEEE Std 802.1Q C-VLAN and MAC Bridge component Reserved addresses (see Table 8-1 in IEEE Std 802.1Q-2014), the S-VLAN component Reserved addresses (see Table 8-2 in IEEE Std 802.1Q-2014), and the TPMR component Reserved addresses (see Table 8-3 in IEEE Std 802.1Q-2014). The choice of address used determines the scope of propagation of LLDPDUs within a bridged LAN, as follows:

a) The ***nearest bridge*** group MAC address is an address that no conformant Two-Port MAC Relay (TPMR) component, S-VLAN component, C-VLAN component, or MAC Bridge component can forward. LLDPDUs transmitted using this destination address can therefore travel no further than those stations that can be reached via a single individual LAN from the originating station. LLDPDUs received on this address therefore represent information about stations that are attached to the same individual LAN segment as the recipient station.

NOTE 1—This address was selected in order to make it possible to transmit an LLDP frame containing information specific to a single individual LAN, and for that information not to be propagated further than the extent of that individual LAN, to avoid the meaning of that information being misinterpreted. For example, a TLV containing the auto-negotiation status of an IEEE 802.3™ LAN would be open to misinterpretation if it were to be propagated via a Bridge

onto a second IEEE 802.3 LAN, and would be meaningless if propagated onto a LAN based on a different MAC technology. This is the "LLDP_Multicast address" that was used in IEEE Std 802.1AB-2005. This address was erroneously omitted from the table of S-VLAN component Reserved addresses added to IEEE Std 802.1Q by IEEE Std 802.1ad-2005, but has been added to that table by IEEE Std 802.1aj™-2009 [B1] and is included in subsequent revisions of IEEE Std 802.1Q.

b) The ***nearest non-TPMR bridge*** group MAC address is an address that no conformant C-VLAN component, S-VLAN component, or MAC Bridge can forward. However, this address is relayed by TPMR components. Therefore, LLDPDUs received on this address represent information about stations that are not separated from the recipient station by any intervening C-VLAN component, S-VLAN component, or MAC Bridge. There may, however, be one or more TPMR components in the path between the originating and receiving stations.

NOTE 2—This address was selected in order to make it possible to communicate information between adjacent bridges and for that communication to be transparent to the presence or absence of TPMRs in the transmission path. This address is primarily intended for use within provider bridged networks.

c) The ***nearest Customer Bridge*** group MAC address is an address that no conformant C-VLAN component or MAC Bridge forwards. However, this address is relayed by TPMR components and S-VLAN components. Therefore, LLDPDUs received on this address represent information about stations that are not separated from the recipient station by any intervening C-VLAN components (e.g., a C-VLAN Bridge or Provider Edge Bridge). There may, however, be TPMR components and/or S-VLAN components in the path between the originating and receiving stations.

NOTE 3—This address was selected in order to make it possible to communicate information between adjacent Customer Bridges and for that communication to be transparent to the presence or absence of TPMRs or S-VLAN components in the communication path. The scope of this address is the same as that of a customer-to-customer MACSec connection.

Table 7-1—Group MAC addresses used by LLDP

Name	Value	Purpose
Nearest bridge	01-80-C2-00-00-0E	Propagation constrained to a single physical link; stopped by all types of bridge
Nearest non-TPMR bridge	01-80-C2-00-00-03	Propagation constrained by all bridges other than TPMRs; intended for use within provider bridged networks
Nearest Customer Bridge	01-80-C2-00-00-00	Propagation constrained by customer bridges; this gives the same coverage as a customer-customer MACSec connection

NOTE 4—The LSAP or EtherType field is necessary to distinguish between different protocols conveyed in frames with the same group or individual destination address. Prior to revision of this standard to allow multiple address scopes, the destination address 01-80-C2-00-00-00 was only explicitly specified for Spanning Tree Protocol BPDUs. It is therefore possible that some implementations recognize only a frame's destination address before applying priority handling resources intended to guard against BPDU loss. Since LLDP rate limits LLDPDU transmission, the additional consumption of these resources by LLDPDUs is unlikely to pose a problem for robust implementations.

It is assumed in the foregoing definitions of the scope of these group MAC addresses that an end station attached to an individual LAN is a potential recipient of any of these addresses, and therefore such end stations fall within the scope of these addresses as well as the identified bridge components.

In addition to determining the scope of transmission, the support of more than one destination MAC address allows different sets of TLVs to be supported over the different transmission scopes.

In addition to the prescribed support for standard group MAC addresses shown in Table 7-1, implementations of LLDP may support the following destination addresses for LLDPDUs:

d) Any group MAC address.
e) Any individual MAC address.

Support for the use of each of these destination addresses, for both transmission and reception of LLDPDUs, is either mandatory, recommended, permitted, or not permitted, according to the type of system in which LLDP is implemented, as shown in Table 7-2.

Table 7-2—Support for MAC addresses in different systems

Address	C-VLAN Bridge	S-VLAN Bridge	TPMR Bridge	End station
Nearest bridge	Mandatory	Mandatory	Mandatory	Mandatory
Nearest non-TPMR bridge	Mandatory	Mandatory	Not permitted	Recommended
Nearest Customer Bridge	Mandatory	Not permitted	Not permitted	Recommended
Any other group MAC address	Permitted	Permitted	Permitted	Permitted
Any individual MAC address	Permitted	Permitted	Permitted	Permitted

NOTE 5 —Where the implementation supports the use of individual MAC addresses, there is a need for some means whereby the sender can ascertain what address to use, and what scope is associated with that address; how this is achieved is outside the scope of this standard.

NOTE 6—The option of using an individual MAC addresses supports the use of LLDP in IEEE Std 802.11™ [B2] and other wireless LANs, where it is desirable to target specific TLV content at specific logical Ports. In particular, the use of an individual MAC address allows an access point to direct LLDP frames at an individual station with which it is associated.

NOTE 7—If an individual MAC address is used as the destination address by a given LLDP Agent, then that agent would not also receive LLDPDUs on that address; it makes no sense for an LLDP Agent to send LLDP frames to an individual MAC address that it also receives LLDP frames on. Similarly, if an Agent is able to receive LLDPDUs sent to a given individual MAC address, it would not also transmit LLDPDUs to that address. Therefore, Agents associated with individual MAC addresses are either transmitters or receivers of LLDPDUs for a given address, but not both.

NOTE 8—The destination MAC address used by a given LLDP agent defines only the scope of transmission and the intended recipient(s) of the LLDPDUs; it plays no part in protocol identification. In particular, the group MAC addresses identified in Table 7-1 are not used exclusively by LLDP; other protocols that require to use a similar transmission scope are free to use the same addresses.

7.2 Source address

The source address shall be the individual MAC address of the sending station or port.

7.3 EtherType use and encoding

The EtherType used to identify the LLDP protocol shall be the LLDP EtherType specified in Table 7-3.

Table 7-3—LLDP EtherType

Name	Value
LLDP EtherType	88-CC

Where the LLC entity uses an MSAP that is supported by a specific media access control method (for example, IEEE Std 802.3) or a media access control independent entity (for example, IEEE Std 802.1AE) that directly supports encoding of EtherTypes, the LLC entity shall encode the LLDP EtherType as the two octet LLDPDU header in the MSDU of the corresponding MAC service request.

Where the LLC entity uses an MSAP that is supported by a specific media access control method that does not directly support Ethertype encoding (for example, IEEE Std 802.11 [B2]), the encoding shall be as specified in IEEE Std 802.1AC.

NOTE—Annex C provides example LLDP transmission frame formats for both direct-encoded and SNAP-encoded LLDP EtherType encoding methods.

7.4 LLDPDU reception

The LLDPDU shall be delivered to the LLDP receive module if, and only if

a) The destination MAC address is equal to the MAC address associated with the corresponding LLDP Agent; and

NOTE—The destination MAC address can be one of the addresses in Table 7-1, or any other valid MAC address—see 7.1.

b) The EtherType is equal to the value shown in Table 7-3.

8. LLDPDU and TLV formats

8.1 LLDPDU bit and octet ordering conventions

All LLDPDUs shall contain an integral number of octets. The octets in an LLDPDU are numbered starting from 1 and increasing in the order they are put into the LLDPDU. The bits in an octet are numbered from 1 to 8, where bit 1 is the low-order bit.

When consecutive bits within an octet are used to represent a binary number, the highest bit number has the most significant value. When consecutive octets are used to represent a binary number, the lower octet number has the most significant value. All TLVs respect these bit and octet ordering conventions.

When the encoding of a field or a number of fields is represented using a diagram

a) Octets are shown with the lowest numbered octet nearest the left of the page, the octet numbering increasing from left to right.
b) Within an octet, bits are shown with bit 8 to the left and bit 1 to the right.

8.2 LLDPDU format

The LLDPDU shall contain the following ordered sequence of three mandatory TLVs followed by zero or more optional TLVs as shown in Figure 8-1. An End of LLDPDU TLV may be present as the last TLV in the LLDPDU.

a) Three mandatory TLVs shall be included at the beginning of each LLDPDU and shall be in the order shown.
    1) Chassis ID TLV
    2) Port ID TLV
    3) Time To Live TLV
b) Optional TLVs as selected by network management (may be inserted in any order).

NOTE 1—”Optional” in the sense that they are not required for LLDP operation; however, their presence could be required by other system elements that use LLDP.

c) If the End Of LLDPDU TLV is present, it shall be the last TLV in the LLDPDU.

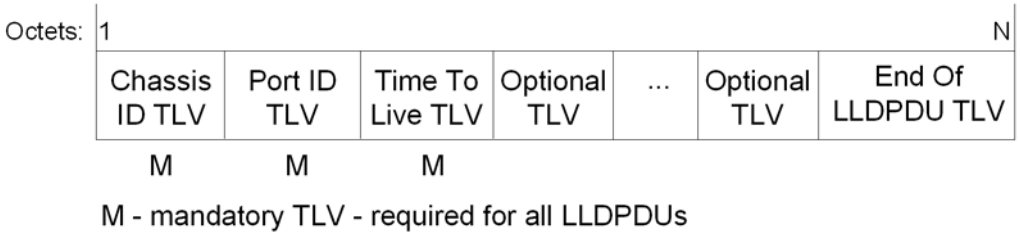

Figure 8-1—LLDPDU format

The maximum length of the LLDPDU shall be the maximum information field length allowed by the particular transmission rate and protocol. In IEEE 802.3 MACs, for example, the maximum LLDPDU length is the maximum data field length for the basic, untagged MAC frame (1500 octets).

NOTE 2—There is no defined minimum length of an LLDPDU, other than that implied by the requirement that conformant implementations support the mandatory TLVs specified in Table 8-1.

8.3 TLV categories

The TLVs are grouped into two general categories as follows:

a) A set of TLVs that are considered to be basic to the management of network stations and that are a required capability of all LLDP implementations. Each TLV in this category is identified by a unique TLV type value that indicates the particular kind of information contained in the TLV.
b) Organizationally specific extension sets of TLVs that are defined by standards groups such as IEEE 802.1 and IEEE 802.3 and others to enhance management of network stations that are operating with particular media and/or protocols.
    1) TLVs in this category are identified by a common TLV type value that indicates the TLV as belonging to the set of Organizationally Specific TLVs.
    2) Each organization is identified by its organizationally unique identifier (OUI).
    3) Organizationally Specific TLV subtype values indicate the kind of information contained in the TLV.

The basic TLV format and general field definition rules are defined in 8.4. Specific definitions and usage requirements for all basic management set TLVs are defined in 8.5. Usage rules/requirements that pertain to each individual basic TLV are contained in the definition for that particular TLV.

The basic format and the field definition/general usage rules/restrictions for Organizationally Specific TLVs are defined in 8.6. Usage rules/requirements that pertain to each individual basic TLV are contained in the definition for that particular TLV.

8.4 Basic TLV format

Figure 8-2 shows the basic TLV format.

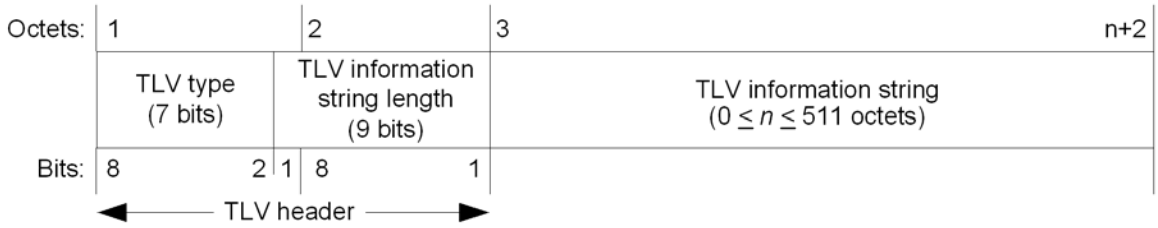

Figure 8-2—Basic TLV format

The TLV type field occupies the seven most significant bits of the first octet of the TLV format. The least significant bit in the first octet of the TLV format is the most significant bit of the TLV information string length field.

8.4.1 TLV type

The TLV type field is seven bits long and identifies the specific TLV. Two classes of TLVs are defined:

a) Mandatory TLVs that shall be included in all LLDPDUs.
b) Optional TLVs that may be included in LLDPDUs.

Table 8-1 lists the currently defined TLVs, their identifying TLV type values, and whether they are mandatory or optional for inclusion in any particular LLDPDU.

Table 8-1—TLV type values

TLV type [a]	TLV name	Usage in LLDPDU	Reference
0	End Of LLDPDU	Optional	8.5.1
1	Chassis ID	Mandatory	8.5.2
2	Port ID	Mandatory	8.5.3
3	Time To Live	Mandatory	8.5.4
4	Port Description	Optional	8.5.5
5	System Name	Optional	8.5.6
6	System Description	Optional	8.5.7
7	System Capabilities	Optional	8.5.8
8	Management Address	Optional	8.5.9
9–126	Reserved for future standardization	—	—
127	Organizationally Specific TLVs	Optional	—

[a]TLVs with type values 0–8 are members of the basic management set.

8.4.2 TLV information string length

The TLV information string length field shall contain the length of the information string, in octets.

8.4.3 TLV information string

The information string

a) May be fixed or variable length.
b) May include one or more information fields with associated subtype identifiers and field length designators as in, for example, the Management Address TLV (see 8.5.9).
c) May contain either binary or alpha-numeric information that is instance specific for the particular TLV type and/or subtype.
    1) Bit 1 in binary bit maps shall be the least significant bit in the field.
    2) The first octet of an alpha-numeric field shall be the most significant octet.
    3) Alpha-numeric information shall be encoded in UTF-8 [IETF RFC 3629].

8.5 Basic management TLV set formats and definitions

8.5.1 End Of LLDPDU TLV

The End Of LLDPDU TLV is a 2-octet, all-zero TLV that is used to mark the end of the TLV sequence in LLDPDUs. The format for this TLV is shown in Figure 8-3.

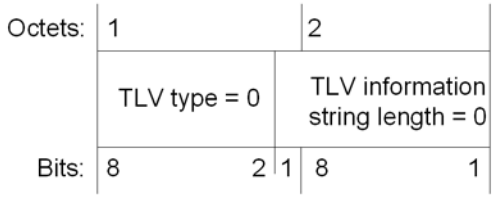

Figure 8-3—End Of LLDPDU TLV format

NOTE—Some IEEE 802 MACs require the data field in a frame to contain a minimum number of octets. For example, the IEEE 802.3 MAC adds pad octets to complete a minimum length data field if the user's data is less than the minimum required length. Since pad octets are unspecified, an End Of LLDPDU TLV can be used to prevent non-zero pad octets from being interpreted by the receiving LLDP agent as another TLV.

8.5.2 Chassis ID TLV

The Chassis ID TLV is a mandatory TLV that identifies the chassis containing the IEEE 802 LAN station associated with the transmitting LLDP agent. There are several ways in which a chassis may be identified and a chassis ID subtype is used to indicate the type of component being referenced by the chassis ID field. Each LLDPDU shall contain one, and only one, Chassis ID TLV and the chassis ID field value shall remain constant for all LLDPDUs while the MAC status parameter for the Port remains operational.

The Chassis ID TLV shall be the first TLV in the LLDPDU. Its format is shown in Figure 8-4.

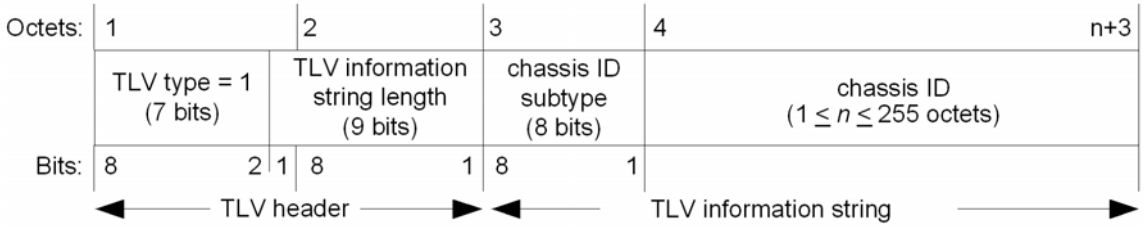

Figure 8-4—Chassis ID TLV format

8.5.2.1 TLV information string length

The TLV information string length field shall indicate the exact length, in octets, of the (chassis ID subtype + chassis ID) fields.

8.5.2.2 chassis ID subtype

The chassis ID subtype field shall contain an integer value indicating the basis for the chassis ID entity that is listed in the chassis ID field. The defined chassis ID subtypes and their preferred use order are listed in Table 8-2.

Table 8-2—chassis ID subtype enumeration

ID subtype	ID basis	Reference
0	Reserved	—
1	Chassis component	EntPhysicalAlias when entPhysClass has a value of ‘chassis(3)’ (IETF RFC 6933)
2	Interface alias	IfAlias (IETF RFC 2863)
3	Port component	EntPhysicalAlias when entPhysicalClass has a value ‘port(10)’ or ‘backplane(4)’ (IETF RFC 6933)
4	MAC address	MAC address (IEEE Std 802)
5	Network address	networkAddress [a]
6	Interface name	ifName (IETF RFC 2863)
7	Locally assigned	local[b]
8–255	Reserved	—

[a]networkAddress is an octet string that identifies a particular network address family and an associated network address that are encoded in network octet order. An IP address, for example, would be encoded with the first octet containing the IANA Address Family Numbers enumeration value for the specific address type and octets 2 through *n* containing the address value (for example, the encoding for C0-00-02-0A would indicate the IPv4 address 192.0.2.10).
[b]local is an alpha-numeric string and is locally assigned.

8.5.2.3 chassis ID

The chassis ID field shall contain an octet string indicating the specific identifier for the particular chassis in this system. Because chassis ID and port ID values are concatenated to form the local MSAP identifier, the value chosen from Table 8-2 for the chassis ID shall be non-null.

8.5.2.4 Chassis ID TLV usage rules

An LLDPDU shall contain exactly one Chassis ID TLV.

8.5.3 Port ID TLV

The Port ID TLV is a mandatory TLV that identifies the port component of the MSAP identifier associated with the transmitting LLDP agent. As with the chassis, there are several ways in which a port may be identified. A port ID subtype is used to indicate how the port is being referenced in the port ID field. Each LLDPDU shall contain one, and only one, Port ID TLV. The port ID value shall remain constant for all LLDPDUs while the transmitting port remains operational.

The chosen port shall be identified in an unambiguous manner (for example, backplane number, management entity, MAC address).

The Port ID TLV shall be the second TLV in the LLDPDU. Its format is shown in Figure 8-5.

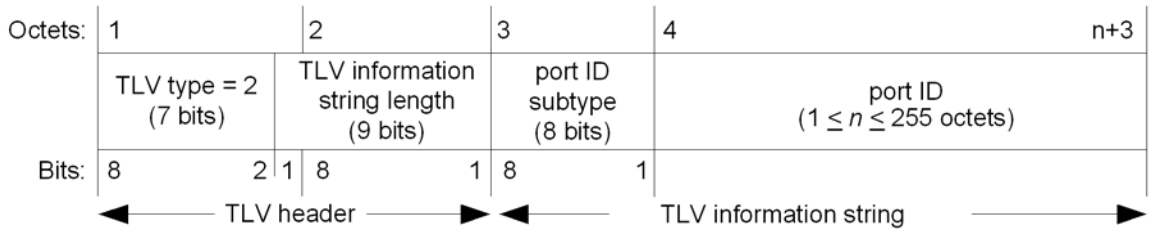

Figure 8-5—Port ID TLV format

8.5.3.1 TLV information string length

The TLV information string length field shall indicate the length, in octets, of the (port ID subtype + port ID) fields.

8.5.3.2 port ID subtype

The port ID subtype field shall contain an integer value indicating the basis for the identifier that is listed in the port ID field. The defined port ID subtypes and their preferred use order are listed in Table 8-3.

Table 8-3—port ID subtype enumeration

ID subtype	ID basis	References
0	Reserved	—
1	Interface alias	ifAlias (IETF RFC 2863)
2	Port component	entPhysicalAlias when entPhysicalClass has a value ‘port(10)’ or ‘backplane(4)’ (IETF RFC 6933)
3	MAC address	MAC address (IEEE Std 802)
4	Network address	networkAddress [a]
5	Interface name	ifName (IETF RFC 2863)
6	Agent circuit ID	agent circuit ID (IETF RFC 3046)
7	Locally assigned	local [b]
8–255	Reserved	—

[a]networkAddress is an octet string that identifies a particular network address family and an associated network address that are encoded in network octet order. An IP address, for example, would be encoded with the first octet containing the IANA Address Family Numbers enumeration value for the specific address type and octets 2 through *n* containing the address value (for example, the encoding for C0-00-02-0A would indicate the IP version 4 address 192.0.2.10).

[b]local is an alpha-numeric string and is locally assigned.

8.5.3.3 port ID

The port ID field is an alpha-numeric string that contains the specific identifier for the port from which this LLDPDU was transmitted. Because chassis ID and port ID values are concatenated to form the local MSAP identifier, the value chosen from Table 8-3 for the port ID shall be non-null.

NOTE—The port ID can contain alphanumeric data or binary data, depending upon the value of the port ID subtype.

8.5.3.4 Port ID TLV usage rules

An LLDPDU shall contain exactly one Port ID TLV.

8.5.4 Time To Live TLV

The Time To Live TLV indicates the number of seconds that the recipient LLDP agent is to regard the information associated with this MSAP identifier to be valid.

a) When the TTL field is non-zero the receiving LLDP agent is notified to completely replace all information associated with this MSAP identifier with the information in the received LLDPDU.
b) When the TTL field is set to zero, the receiving LLDP agent is notified to delete all system information associated with the LLDP agent/port. This TLV may be used, for example, to signal that the sending port has initiated a port shutdown procedure.

The Time To Live TLV is mandatory and shall be the third TLV in the LLDPDU. Its format is shown in Figure 8-6.

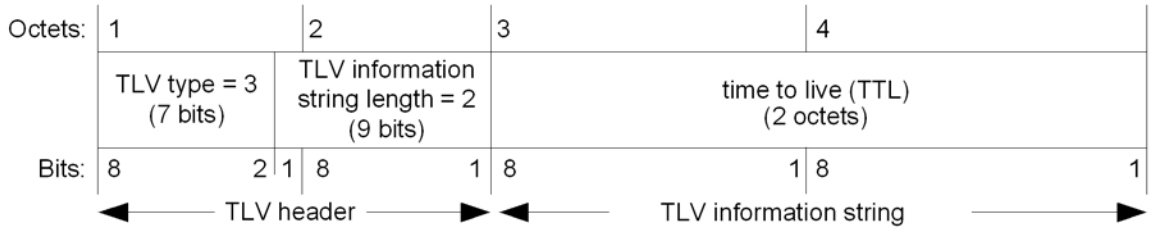

Figure 8-6—Time To Live TLV format

8.5.4.1 time to live (TTL)

The TTL field shall contain an integer value in the range $0 \geq t \geq 65535$ seconds and is set to the computed value of txTTL at the time the LLDPDU is constructed (see 9.2.5.22).

8.5.4.2 Time To Live TLV usage rules

An LLDPDU shall contain exactly one Time To Live TLV.

8.5.5 Port Description TLV

The Port Description TLV allows network management to advertise the IEEE 802 LAN station's port description. The format for this TLV is shown in Figure 8-7.

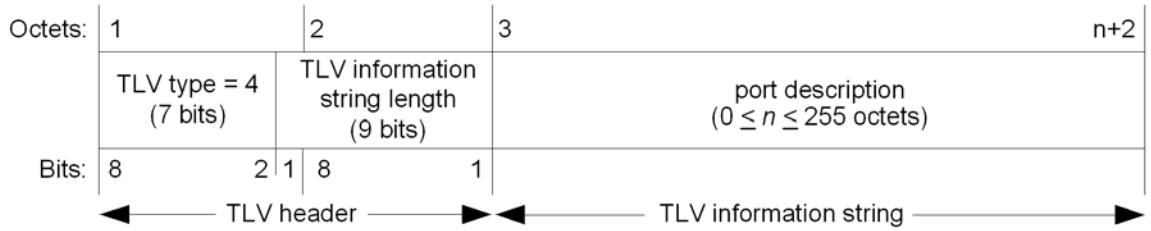

Figure 8-7—Port Description TLV format

8.5.5.1 TLV information string length

The TLV information string length field shall contain the exact length, in octets, of the port description field.

8.5.5.2 port description

The port description field shall contain an alpha-numeric string that indicates the port's description. If IETF RFC 2863 is implemented, the ifDescr object should be used for this field.

8.5.5.3 Port Description TLV usage rules

An LLDPDU should not contain more than one Port Description TLV.

8.5.6 System Name TLV

The System Name TLV allows network management to advertise the system's assigned name. The format for this TLV is shown in Figure 8-8.

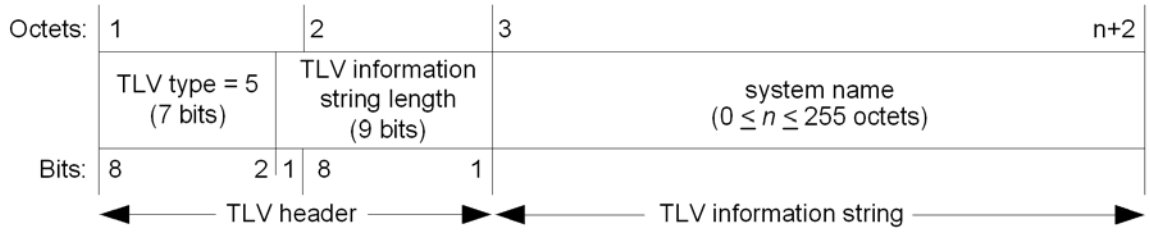

Figure 8-8—System Name TLV format

8.5.6.1 TLV information string length

The TLV information string length field shall contain the exact length, in octets, of the system name.

8.5.6.2 system name

The system name field shall contain an alpha-numeric string that indicates the system's administratively assigned name. The system name should be the system's fully qualified domain name. If implementations support IETF RFC 3418, the sysName object should be used for this field.

8.5.6.3 System Name TLV usage rules

An LLDPDU shall not contain more than one System Name TLV.

8.5.7 System Description TLV

The System Description TLV allows network management to advertise the system's description. The format for this TLV is shown in Figure 8-9.

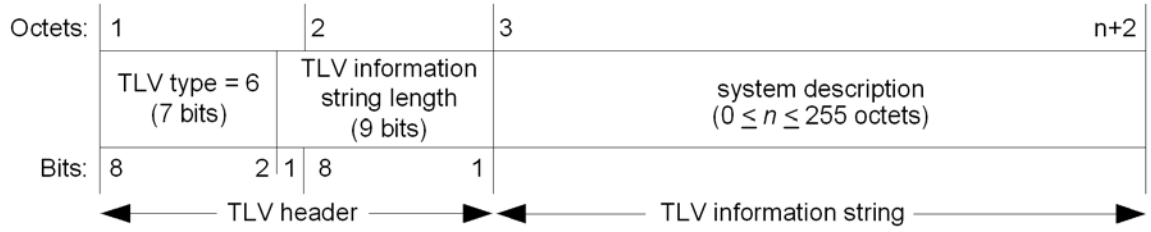

Figure 8-9—System Description TLV format

8.5.7.1 TLV information string length

The TLV information string length field shall indicate the exact length, in octets, of the system description.

8.5.7.2 system description

The system description field shall contain an alpha-numeric string that is the textual description of the network entity. The system description should include the full name and version identification of the system's hardware type, software operating system, and networking software. If implementations support IETF RFC 3418, the sysDescr object should be used for this field.

8.5.7.3 System Description TLV usage rules

An LLDPDU shall not contain more than one System Description TLV.

8.5.8 System Capabilities TLV

The System Capabilities TLV is an optional TLV that identifies the primary function(s) of the system and whether or not these primary functions are enabled. Figure 8-10 shows the format of this TLV.

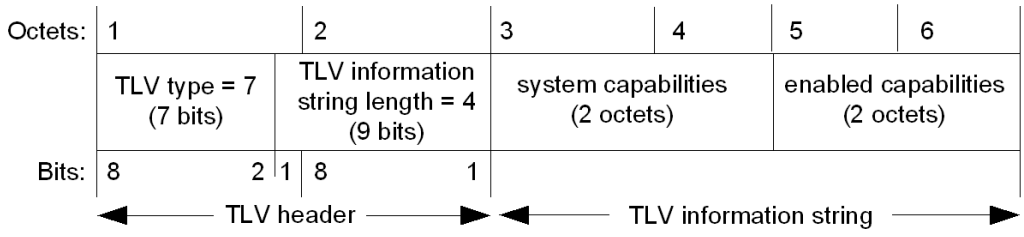

Figure 8-10—System Capabilities TLV format

8.5.8.1 system capabilities

The system capabilities field shall contain a bit-map of the capabilities that define the primary function(s) of the system. The bit positions for each function and the associated MIB or standard that are likely, but not guaranteed, to be supported are listed in Table 8-4. A binary one in the associated bit indicates the existence of that capability. Individual systems may indicate more than one implemented functional capability (for example, both a bridge and router capability).

Table 8-4—System capabilities

Bit	Capability	Reference
1	Other	—
2	Repeater	IETF RFC 2108 [B7]
3	MAC Bridge component	IEEE Std 802.1Q
4	802.11 Access Point (AP)	IEEE Std 802.11 MIB
5	Router	IETF RFC 1812 [B6]
6	Telephone	IETF RFC 4293 [B10]
7	DOCSIS cable device	IETF RFC 4639 and IETF RFC 4546 [B12]
8	Station Only [a]	IETF RFC 4293 [B10]
9	C-VLAN component	IEEE Std 802.1Q
10	S-VLAN component	IEEE Std 802.1Q
11	Two-port MAC Relay component	IEEE Std 802.1Q
12–16	reserved	—

[a]The Station Only capability is intended for devices that implement only an end station capability, and for which none of the other capabilities in the table apply. Bit 8 should therefore not be set in conjunction with any other bits.

8.5.8.2 enabled capabilities

The enabled capabilities field shall contain a bit map of the primary functions listed in Table 8-4. A binary one in a bit position indicates that the function associated with that bit is currently enabled.

8.5.8.3 System Capabilities TLV usage rules

An LLDPDU shall not contain more than one System Capabilities TLV.

If the system capabilities field does not indicate the existence of a capability that the enabled capabilities field indicates is enabled, the TLV is interpreted as containing an error and shall be discarded.

8.5.9 Management Address TLV

The Management Address TLV identifies an address associated with the local LLDP agent that may be used to reach higher layer entities to assist discovery by network management. The TLV also provides room for the inclusion of both the system interface number and an object identifier (OID) that are associated with this management address, if either or both are known. Figure 8-11 shows the Management Address TLV format.

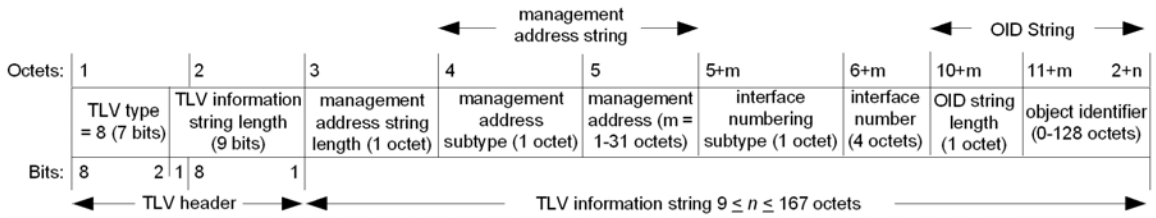

Figure 8-11—Management Address TLV format

8.5.9.1 TLV information string length

The TLV information string length field shall contain the length, in octets, of all the fields in the TLV information string.

8.5.9.2 management address string length

The management address string length field shall contain the length, in octets, of the (management address subtype + management address) fields.

8.5.9.3 management address subtype

The management address subtype field shall contain an integer value indicating the type of address that is listed in the management address field. Enumeration for this field is contained in the ianaAddressFamilyNumbers module of the IETF RFC 3232 on-line database that is accessible through a web page (http://www.iana.org). The management address subtype is contained in the first octet of the management address string.

NOTE—As a consequence of the limitation on the size of the management address field (a maximum of 31 octets), some options for the management address subtype (such as long DNS names) may be impractical.

8.5.9.4 management address

The management address field shall contain an octet string indicating the particular management address associated with this TLV.

a) The returned address should be the most appropriate for management use, typically a layer 3 address such as the IPv4 address, 192.0.2.10 (see also Table 8-2, footnote a).
b) If no management address is available, the return address should be the MAC address for the station or port. The ianaAddressFamilyNumber for a MAC address is all802 (value 6).

8.5.9.5 interface numbering subtype

The interface numbering subtype field shall contain an integer value indicating the numbering method used for defining the interface number. The following three values are currently defined:

1) Unknown
2) ifIndex
3) system port number

8.5.9.6 interface number

The interface number field shall contain the assigned number within the system that identifies the specific interface associated with this management address. If the value of the interface subtype is unknown, this field shall be set to zero.

8.5.9.7 object identifier (OID) string length

The object identifier string length field shall contain the length, in octets, of the OID. A value of zero in this field indicates that the OID field is not provided.

8.5.9.8 object identifier

The object identifier field contains an OID that identifies the type of hardware component or protocol entity associated with the indicated management address. The OID shall be the value portion of the ASN.1 encoding of the object identifier, encoded according to ASN.1 basic encoding rules [ISO/IEC 8824-1]. If no OID is available, this field shall not be provided.

NOTE—The interface number and OID are included in this TLV to assist NMS discovery by indicating Enterprise Specific or other starting points for the search, such as the Interface or Entity MIB (IETF RFC 6933).

8.5.9.9 Management Address TLV usage rules

Management Address TLVs are subject to the following:

a) At least one Management Address TLV should be included in every LLDPDU.
b) Since there are typically a number of different addresses associated with a MSAP identifier, an individual LLDPDU may contain more than one Management Address TLV.
c) When Management Address TLV(s) are included in an LLDPDU, the included address(es) should be the address(es) offering the best management capability.
d) If more than one Management Address TLV is included in an LLDPDU, each management address shall be different from the management address in any other management address TLV in the LLDPDU.
e) If an OID is included in the TLV, it shall be reachable by the management address.
f) In a properly formed Management Address TLV, the TLV information string length is equal to:

   (management address string length) + (OID string length) + 7.

   If the TLV information string length in a received Management Address TLV is incorrect, then it is ignored and processing of that LLDPDU is terminated.

8.6 Organizationally Specific TLVs

This TLV category is provided to allow different organizations, such as IEEE 802.1, IEEE 802.3, IETF, as well as individual software and equipment vendors, to define TLVs that advertise information to remote entities attached to the same media, subject to the following restrictions:

a) Information transmitted in an Organizationally Specific TLV is intended to be a one way advertisement.
b) Information transmitted in an Organizationally Specific TLV shall be independent from information in a TLV received from a remote port.
c) Information transmitted in one Organizationally Specific TLV shall not be concatenated with information transmitted in another TLV on the same media in order to provide a means for sending messages that are larger than would fit within a single TLV.

d) Information received in an Organizationally Specific TLV shall not be explicitly forwarded to other ports in the system.
e) Organizationally Specific TLVs shall conform to 8.4 and 8.6.1.

Each set of Organizationally Specific TLVs shall include associated LLDP MIB extensions and the associated TLV selection management variables and MIB/TLV cross reference tables. Systems that implement LLDP and that also support standard protocols for which Organizationally Specific TLV extension sets have been defined shall support all TLVs and the LLDP MIB extensions defined for that particular TLV set.

Annex D of IEEE Std 802.1Q-2014 and Clause 79 of IEEE Std 802.3-2012 contain Organizationally Specific TLVs for IEEE Std 802.1 and IEEE Std 802.3, respectively, along with their associated LLDP MIB extensions and LLDP MIB/TLV cross reference tables. Organizations wishing to define TLVs for their use should use these annexes as examples.

8.6.1 Basic Organizationally Specific TLV format

The basic format for Organizationally Specific TLVs is shown in Figure 8-12.

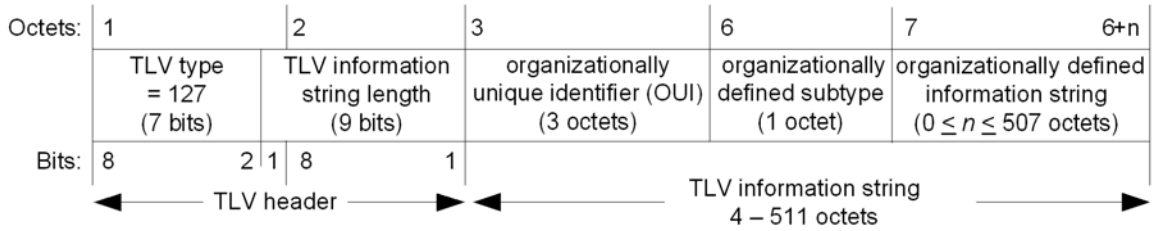

Figure 8-12—Basic format for Organizationally Specific TLVs

The following subclauses indicate how the individual fields shall be defined.

8.6.1.1 TLV type

The same Organizationally Specific TLV type value, 127, shall be used for all organizationally defined TLVs.

8.6.1.2 TLV information string length

The TLV information string length field shall contain the length of the information string, in octets.

8.6.1.3 organizationally unique identifier (OUI)

The organizationally unique identifier field shall contain the organization's OUI (see Clause 8 of IEEE Std 802-2014).

8.6.1.4 organizationally defined subtype

The organizationally defined subtype field shall contain a unique subtype value assigned by the defining organization.

NOTE—Defining organizations are responsible for maintaining listings of organizationally defined subtypes in order to assure uniqueness.

8.6.1.5 organizationally defined information string

The actual format of the organizationally defined information string field is organizationally specific and can

a) Contain either binary or alpha-numeric information that is instance specific for the particular TLV type and/or subtype. Alpha-numeric information should be encoded in UTF-8 (IETF RFC 3629).
b) Include one or more information fields with their associated field-type identifiers and field length designators similar to those in the Management Address TLV (see 8.5.9).

8.6.2 Organizationally Specific TLV usage rules

Organizations defining their own Organizationally Specific TLVs should include a subclause that defines any specific usage rules and/or specific conditions that affects how the receiving LLDP agent shall treat the TLV.

The Organizationally Specific TLV usage rules should include the following:

a) The number of Organizationally Specific TLVs that may be contained in an LLDPDU and any additional information field subtypes that would identify differences between two TLVs with the same OUI and organizationally defined subtype.
b) Any error conditions that are specific to the particular Organizationally Specific TLV and the action that is taken for each defined error condition.

9. LLDP agent operation

This clause describes the operation of the protocol from the point of view of a single LLDP agent (see Clause 6); i.e., it describes the operation of the protocol for the transmission and reception of LLDPDUs on a single Port, using a MAC address (see Clause 7) as the destination address. Where the use of more than one MAC address is supported on a given Port, a separate instance of the LLDP agent is responsible for the operation of the protocol for each MAC address that is supported.

9.1 Overview

Each LLDP agent is responsible for causing the following tasks to be performed:

a) Maintaining current information in the LLDP local system MIB.
b) Extracting and formatting LLDP local system MIB information for transmission to the remote port LLDP agent(s) at regular intervals or whenever there is a change in the system condition or status.
c) Recognizing and processing received LLDP frames.
d) Maintaining current values in the LLDP remote system MIB.
e) Using the procedure somethingChangedLocal() and the procedure somethingChangedRemote() to notify the PTOPO MIB manager and MIB managers of other optional MIBs whenever a value or status change has occurred in one or more objects in the LLDP local system LLDP remote systems MIB.

NOTE—A consequence of the support of multiple MAC addresses on a Port is that there are multiple copies of the LLDP local system MIB and remote system MIB, one copy for each address supported.

LLDP allows implementations that support three different operating modes: transmit only, receive only, or both transmit and receive. An LLDP agent shall conform to the specifications of each of the state machines indicated in Table 9-1 for the operating mode that it supports.

Table 9-1—Subclause/operating mode applicability

Subclause number	Subclause title	Transmit only	Receive only	Transmit and receive
9.2.9	Receive state machine	—	Mandatory	Mandatory
9.2.8	Transmit state machine	Mandatory	—	Mandatory
9.2.10	Transmit timer state machine	Mandatory	—	Mandatory

9.1.1 Frame transmission

LLDPDU transmission is performed by the transmit state machine, and the timing of transmissions is controlled by the transmit timer state machine. Transmissions occur as a result of a number of different local events, as follows:

a) A regular background transmission occurs as a result of a transmission timer expiring. This ensures that the receiving system does not time out the information received, as a result of the “time to live” associated with the last received LLDPDU expiring.
b) If a *new neighbor* is recognized (i.e., an LLDPDU is received by the receive state machine from a hitherto unknown remote agent), then four LLDPDU transmissions are performed at shorter time intervals to ensure that the new neighbor is quickly updated with current information about its own neighbors. This rapid transmission behavior is suppressed if the remote systems MIB is not able to

accommodate the new neighbor information without removing current information relating to an older neighbor, in order to prevent excessive PDU transmissions resulting from "churn" effects.

c) If a condition (status or value) change occurs in one or more objects in the LLDP local system MIB, then an LLDPDU is transmitted immediately, in order to communicate the local changes as quickly as possible to any neighboring systems. The transmit timer state machine operates a simple credit-based transmission strategy that allows a number of LLDPDUs to be transmitted in quick succession (the maximum number being determined by the maximum credit value), thus allowing rapid updating of information in situations where the local state is changing quickly. However, once the LLDP Agent's transmission credit is exhausted, the rate of transmission is cut to a maximum of one transmission per second.

d) The overall timing of regular transmissions is governed by a system "tick" that operates on a nominal timebase of 1 s. However, in shared media LANs, in order to avoid bunching of transmissions, the intervals between ticks include a random jitter component so that, while the average time interval between ticks is 1 s, the interval between adjacent pairs of ticks can vary (see 9.2.2).

In order to minimize synchronization effects, it is recommended that transmission intervals for different LLDP agents on the same multi-port implementation are staggered.

9.1.2 LLDPDU types

The following two types of LLDPDUs are defined:

a) Normal LLDPDUs that provide management information about the local station to that station's neighbors.

b) A special shutdown advisory LLDPDU indicating that any information about the local station that is maintained in a remote LLDP agent's remote systems MIB is now invalid and is to be discarded.

9.1.2.1 Normal LLDPDUs

The LLDP local system MIB contains the information needed for constructing individual TLVs and includes a capability that allows network managers to select the optional TLVs to be included in the LLDPDU (see 10.2.2).

When the transmit state machine (9.2.8) determines that a normal LLDPDU is to be transmitted, the LLDP MIB manager extracts the selected information from the LLDP local system MIB and constructs an LLDPDU as defined in 9.1, containing the following:

a) The mandatory TLVs as specified in 8.2:
   1) Chassis ID TLV (see 8.5.2).
   2) Port ID TLV (see 8.5.3).
   3) Time To Live TLV, with the TTL value set equal to txTTL (see 8.5.4).

b) Additional optional TLVs, from the basic management set or from one or more organizationally specific sets, as allowed by LLDPDU length restrictions and as selected in the LLDP local system MIB by network management (see Table 8-1).

c) Optionally, an End Of LLDPDU TLV.

Optional TLVs from the basic management set, such as the Management Address TLV, with different TLV type and subtype combinations as well as optional TLVs with different OUI and organizationally defined subtype combinations from one or more organizationally specific sets can be included within the same LLDPDU.

9.1.2.2 Shutdown LLDPDUs

A special procedure exists for the case in which a LLDP agent knows an associated port is about to become non-operational (for example, the adminStatus for the port is transitioning to ‘disabled’). In the event a port, currently configured with LLDP frame transmission enabled, either becomes disabled for LLDP activity, or the interface is administratively disabled, the transmit state machine attempts to send a final LLDP shutdown LLDPDU with:

a) The Chassis ID TLV.
b) The Port ID TLV.
c) The Time To Live TLV with the TTL field set to zero.
d) Optionally, an End Of LLDPDU TLV.

The shutdown LLDPDU does not include any other optional TLVs and, if possible, should be transmitted before the interface is disabled.

NOTE—There is an inherent race condition between an interface knowing it is going down and its ability to send “one more frame.” If possible, the actual transition of the port to disabled is delayed until after this frame is transmitted. In the event where adminStatus is transitioning to the disabled state and the LLDP agent is shutting down, this shutdown procedure should be executed for all local ports.

9.1.3 Frame reception

LLDP frame reception consists of three phases: frame recognition, frame validation, and LLDP remote systems MIB updating. All error checking is done during frame validation.

Frame recognition is performed at the LLDP/LSAP (see Figure 6-2), where the destination address and EtherType values determine whether the frame is an LLDP frame destined for this LLDP agent or not. Frames that are recognized as LLDP frames are then validated to determine whether they are properly constructed and contain the correct set of mandatory TLVs; frames that pass the validation criteria are used to update the contents of entries in the LLDP remote systems MIB for the originating LSAP identifier and destination MAC address. If the frame is a normal LLDPDU and an entry does not exist in the LLDP remote systems MIB for the originating LSAP identifier and destination MAC address, then a new entry in the MIB is created, assuming that sufficient space exists to do so. If the frame is a normal LLDPDU and an entry already exists in the remote systems MIB for the originating LSAP identifier and destination MAC address, then the new information contained in the LLDPDU is used to replace the whole of the existing entry in the MIB; i.e., if there are information elements in the existing entry for which there are corresponding elements in the received LLDPDU, then those elements are updated using the information received; any other information elements in the existing entry in the MIB are deleted.

One of the parameters received in an incoming LLDPDU is the TTL value; this determines how long the information associated with that LSAP identifier and destination MAC address is to be stored in the LLDP remote systems MIB before being aged out and deleted. The ageing out of old data ensures that the MIB is purged of data that was originated by systems that are no longer neighbors as a result of topology changes, system failure, or system inactivation.

If the received frame is a shutdown LLDPDU and an entry exists in the LLDP remote systems MIB for the originating LSAP identifier and destination MAC address, then that entry is deleted from the MIB.

9.1.4 Too many neighbors

The amount of space needed in the LLDP remote systems MIB to accommodate the creation of several new MIB structures is beyond the scope of this standard. It may not always be possible to accommodate another

new neighbor in implementations with limited memory. There are several possibilities for handling this case, including but not limited to the following examples:

a) Ignore and not process the new neighbor's information.
b) Delete the information from the oldest neighbor(s) until there is sufficient memory available to store the new neighbor's information.
c) Randomly delete neighbors until there is sufficient memory available to store the new neighbor's information.

The method of handling the case where a new entry in the remote systems MIB can not be created is beyond the scope of this standard, however it is necessary for the implementation to keep track of this case by properly updating the variable tooManyNeighbors and the tooManyNeighborsTimer (9.2.5.15, 9.2.2). The variable tooManyNeighbors identifies when there is insufficient space in the LLDP remote systems MIB to store information from all active neighbors. The tooManyNeighborsTimer indicates the minimum time that this condition exists.

Further detail on the handling of the too many neighbors condition can be found in 9.2.7.7.5.

9.1.5 LLDP remote systems rxInfoTTL timer expiration

If the rxInfoTTL timer (9.2.2.1) associated with an MSAP identifier expires, all information associated with that MSAP identifier is deleted and the procedure somethingChangedRemote() notifies the managers of the PTOPO MIB and other optional MIB modules (see Figure 11-1) that something has changed in the LLDP remote systems MIB associated with that MSAP identifier.

9.1.6 LLDP local port/connection failure

If the local port or the connection to the remote system fails unexpectedly before a shutdown frame is received, the LLDP management entity does not delete objects in the LLDP remote systems MIB pertaining to information received from any MSAP identifier through that local port until the port is re-initialized or the associated rxInfoTTL timer (9.2.2.1) expires.

9.2 State machines

The operation of the protocol is represented by the following three state machines and associated timers to indicate when local system managed object values are to be transmitted to refresh the values in remote LLDP agent's LLDP remote systems MIB and when values in the local LLDP agent's remote systems MIB have become invalid:

a) Transmit state machine (9.2.8).
b) Receive state machine (9.2.9).
c) Transmit timer state machine (9.2.10).

9.2.1 Notational conventions used in state diagrams

State diagrams are used to represent the operation of a function as a group of connected, mutually exclusive states. Only one state of a function can be active at any given time.

Each state is represented in the state diagram as a rectangular box, divided into two parts by a horizontal line. The upper part contains the state identifier, written in uppercase letters. The lower part contains any procedures that are executed on entry to the state.

All permissible transitions between states are represented by arrows, the arrowhead denoting the direction of the possible transition. Labels attached to arrows denote the condition(s) that shall be met in order for the transition to take place. A transition that is global in nature (i.e., a transition that occurs from any of the possible states if the condition attached to the arrow is met) is denoted by an open arrow; i.e., no specific state is identified as the origin of the transition.

On entry to a state, the procedures defined for the state (if any) are executed exactly once, in the order that they appear on the page. Each action is deemed to be atomic; i.e., execution of a procedure completes before the next sequential procedure starts to execute. No procedures execute outside of a state block. On completion of all of the procedures within a state, all exit conditions for the state (including all conditions associated with global transitions) are evaluated continuously until such a time as one of the conditions is met. All exit conditions are regarded as Boolean expressions that evaluate to TRUE or FALSE; if a condition evaluates to TRUE, then the condition is met. When the condition associated with a global transition is met, it supersedes all other exit conditions, including UCT. The label UCT denotes an unconditional transition (i.e., UCT always evaluates to TRUE). The label ELSE denotes a transition that occurs if none of the other conditions for transitions from the state are met (i.e., ELSE evaluates to TRUE if all other possible exit conditions from the state evaluate to FALSE).

A variable that is set to a particular value in a state block retains this value until a subsequent state block executes a procedure that modifies the value, or until the value is modified by an external event or timer tick.

Where it is necessary to segment a state machine description across more than one diagram, a transition between two states that appear on different diagrams is represented by an exit arrow drawn with dashed lines, plus a reference to the diagram that contains the destination state. Similarly, dashed arrows and a dashed state box are used on the destination diagram to show the transition to the destination state. In a state machine that has been segmented in this way, any global transitions that can cause entry to states defined in one of the diagrams are deemed to be potential exit conditions for all of the states of the state machine, regardless of which diagram the state boxes appear in.

Should a conflict exist between the interpretation of a state diagram and either the corresponding global transition tables or the textual description associated with the state machine, the state diagram takes precedence.

The interpretation of the special symbols and operators used in the state diagrams is defined in Table 9-2; these symbols and operators are derived from the notation of the “C” programming language.

Table 9-2—State machine symbols

Symbol	Interpretation
( )	Used to force the precedence of operators in Boolean expressions and to delimit the argument(s) of actions within state boxes.
;	Used as a terminating delimiter for actions within state boxes. Where a state box contains multiple actions, the order of execution follows the normal English language conventions for reading text.
=	Assignment action. The value of the expression to the right of the operator is assigned to the variable to the left of the operator. Where this operator is used to define multiple assignments, e.g., a = b = X the action causes the value of the expression following the right-most assignment operator to be assigned to all of the variables that appear to the left of the right-most assignment operator.
!	Logical NOT operator.

Table 9-2—State machine symbols *(continued)*

Symbol	Interpretation
&&	Logical AND operator.
||	Logical OR operator.
if...then...	Conditional action. If the Boolean expression following the **if** evaluates to TRUE, then the action following the **then** is executed.
{statement 1, ... statement N}	Compound statement. Braces are used to group statements that are executed together as if they were a single statement.
!=	Inequality. Evaluates to TRUE if the expression to the left of the operator is not equal in value to the expression to the right.
==	Equality. Evaluates to TRUE if the expression to the left of the operator is equal in value to the expression to the right.
<	Less than. Evaluates to TRUE if the value of the expression to the left of the operator is less than the value of the expression to the right.
>	Greater than. Evaluates to TRUE if the value of the expression to the left of the operator is greater than the value of the expression to the right.
>=	Greater than or equal to. Evaluates to TRUE if the value of the expression to the left of the operator is either greater than or equal to the value of the expression to the right.
+	Arithmetic addition operator.
–	Arithmetic subtraction operator.

9.2.2 Timers

A set of timers is used by the LLDP state machines; these operate as countdown timers (i.e., they expire when their value reaches zero). These timers

a) Have a resolution of one second.
b) Have an integer value n, where $0 < n < 65535$ seconds.
c) Are started by loading an initial integer value.
d) Are decremented once per timer tick as long as $n > 0$.
e) Represent the remaining time in the period.

For Ports connected to point-to-point LANs, the interval between timer ticks is 1 s. For Ports connected to shared media LANs, the interval between any pair of timer ticks is 0.8 s, plus a “jitter” component that is a random value between 0.0 s and 0.4 s. Hence, the average interval between timer ticks is 1 s; however, the interval between any pair of ticks varies randomly. When a timer tick occurs, all instances of the variable txTick (9.2.5.21) for the Port are set TRUE.

NOTE—The random component of the tick time used in shared media LANs is intended to minimize the possibility of systems connected to the same LAN becoming synchronized in their transmission timings.

9.2.2.1 rxInfoTTL

An instance of this timer exists for each entry in the LLDP remote systems MIB that has been created by this instance of the receive state machine for a given MSAP identifier. The timer value indicates the number of seconds remaining until the information in all the objects in the LLDP remote systems MIB associated with the MSAP identifier is no longer valid and needs to be deleted.

9.2.2.2 tooManyNeighborsTimer

This timer indicates the number of seconds remaining during which it is known that an LLDP neighbor exists that may not be stored in the remote systems MIB.

9.2.2.3 txTTR

An instance of this timer exists for each Agent. The txTTR timer is used to determine the next LLDPDU transmission is due; it is initialized by the transmit timer state machine, with the value msgTxInterval (9.2.5.7) if txFast (9.2.5.18) is equal to zero, or with the value msgFastTx (9.2.5.5) if txFast is greater than zero.

9.2.2.4 txShutdownWhile

An instance of this timer exists for each Agent. The txShutdownWhile timer indicates the number of seconds remaining until LLDP re-initialization can occur.

9.2.3 Per-System variables

An instance of each of these variables exists per system; they are available for use by all of the state machines of all of the LLDP agents in the system.

9.2.3.1 BEGIN

This Boolean variable is controlled by the system initialization process. A value of TRUE causes all state machines to continuously execute their initial state. A value of FALSE allows all state machines to perform transitions out of their initial state, in accordance with the relevant state machine definitions.

9.2.4 Per-Port variables

An instance of each of these variables exists per Port; they are available to all of the state machines associated with a Port.

9.2.4.1 portEnabled

This variable is externally controlled. Its value reflects the operational state of the MAC service supporting the port. Its value is TRUE if the MAC service supporting the Port is in an operable condition; otherwise, it is FALSE.

9.2.5 Per-Agent variables

An instance of each of these variables exists per LLDP agent; they are available to all of the state machines associated with an LLDP agent.

9.2.5.1 adminStatus

This variable indicates whether or not the LLDP agent is enabled. The defined values for this variable are as follows:

a) enabledRxTx: The LLDP agent is enabled for reception and transmission of LLDPDUs. This is the recommended default value.
b) enabledTxOnly: The LLDP agent is enabled for transmission of LLDPDUs only.
c) enabledRxOnly: The LLDP agent is enabled for reception of LLDPDUs only.
d) disabled: The LLDP agent is disabled for both reception and transmission.

9.2.5.2 badFrame

This variable indicates that an incoming LLDPDU was unable to be validated because the LLDPDU failed validation and was discarded.

9.2.5.3 localChange

This variable is set TRUE by the somethingChangedLocal() procedure (see 9.2.7.8) when there is a change in the value of the LLDP local system MIB associated with this LLDP agent. The variable is set FALSE by the operation of the transmit timer state machine (see 9.2.10).

9.2.5.4 MAC address

The MAC address associated with this instance of the Agent, i.e., the destination MAC address that is used when the Agent transmits LLDPDUs, and the destination MAC address that, when present in a received LLDPDU, causes the PDU to be passed to this agent instance for processing.

9.2.5.5 msgFastTx

This variable defines the time interval in timer ticks between transmissions during fast transmission periods (i.e., txFast is non-zero). The recommended default value of msgFastTx is 1; this value can be changed by management to any value in the range 1 through 3600.

9.2.5.6 msgTxHold

This variable is used, as a multiplier of msgTxInterval, to determine the value of txTTL that is carried in LLDP frames transmitted by the LLDP agent. The recommended default value of msgTxHold is 4; this value can be changed by management to any value in the range 1 through 100.

9.2.5.7 msgTxInterval

This variable defines the time interval in timer ticks between transmissions during normal transmission periods (i.e., txFast is zero). The recommended default value for msgTxInterval is 30 s; this value can be changed by management to any value in the range 1 through 3600.

9.2.5.8 newNeighbor

This variable is set TRUE by the rxProcessFrame() procedure (see 9.2.7.7) when information is received from a new neighbor. It is set FALSE by the Transmit timer state machine (9.2.10).

NOTE—The newNeighbor variable is used to force a more rapid regular transmission rate when a new neighbor appears, to ensure that the new neighbor also receives new information from the receiving system.

9.2.5.9 rcvFrame

This variable indicates that an LLDP frame has been recognized by the LLDP LSAP function and is ready to be processed by the LLDP receive state machine. If both of the following conditions are TRUE, the variable rcvFrame is set to TRUE and the frame is made available to the receive state machine for validation and processing:

a) The destination address value is the MAC address associated with this specific LLDP Agent.
b) The EtherType value carried by the frame is the LLDP EtherType defined in Table 7-3.

NOTE—The model that is assumed for reception is that incoming LLDPDUs are presented to all LLDP agents associated with the reception Port, and if the destination MAC address is not one that is associated with a given Agent, that LLDPDU is ignored by that agent. In practice, at most one agent processes any received LLDPDU, as there is only one agent associated with any given destination MAC address on a given reception Port.

9.2.5.10 reinitDelay

This parameter indicates the amount of delay from when adminStatus becomes ‘disabled’ until re-initialization is attempted. The recommended default value for reinitDelay is 2 s.

9.2.5.11 remoteChanges

A Boolean indication of whether the status/value of one or more selected objects in the LLDP remote systems MIB has changed or that all objects associated with a particular MSAP identifier have been deleted. This variable takes the value *(rxInfoAge OR (rxChanges && (rxTTL !=0)))*

9.2.5.12 rxChanges

This variable indicates that the incoming LLDPDU has been received with different TLV values from those currently in the LLDP remote systems MIB associated with the LLDPDU’s MSAP identifier.

9.2.5.13 rxInfoAge

This variable is set TRUE when a timer expiry event occurs for the rxInfoTTL timing counter associated with a particular MSAP identifier in the LLDP remote systems MIB (i.e., the counter transitions from non-zero to zero). It is set FALSE by the operation of the Receive state machine.

9.2.5.14 rxTTL

This variable indicates the time to live value associated with all TLVs received in the current frame.

9.2.5.15 tooManyNeighbors

This Boolean variable indicates that there is insufficient space in the LLDP remote systems MIB to store information from all connected active remote ports (see 9.2.7.7.5).

9.2.5.16 txCredit

The number of consecutive LLDPDUs that can be transmitted at any time. This variable is incremented by the txAddCredit() procedure, up to the maximum value specified by the txCreditMax variable (see 9.2.5.17), and is decremented whenever an LLDPDU is transmitted by the transmit state machine.

9.2.5.17 txCreditMax

The maximum value of txCredit. The recommended default value is 5; this value can be changed by management to any value in the range 1 through 10.

9.2.5.18 txFast

This variable is used as a down counter to count the number of transmissions to be made during a fast transmission period. Fast transmission periods are initiated when a new neighbor is detected, and cause LLDPDU transmissions to take place at shorter time intervals than during normal (steady state) operation of the protocol. The fast transmission period ensures that more than one LLDPDU is transmitted when a new neighbor is detected. The first transmission is immediate; the subsequent transmissions occur at intervals

determined by msgFastTx. The number of transmissions is determined by the value of txFastInit (see 9.2.5.19).

9.2.5.19 txFastInit

This variable is used as the initial value for the txFast variable (see 9.2.5.18). This value determines the number of LLDPDUs that are transmitted during a fast transmission period. The recommended default value of txFastInit is 4; this value can be changed by management to any value in the range 1 through 8.

9.2.5.20 txNow

This variable is used to signal to the transmit state machine that a transmission is required. It is set TRUE by the transmit timer state machine, and set FALSE by the transmit state machine when the transmission has been initiated.

9.2.5.21 txTick

This variable is set TRUE by the system timer tick at one second intervals (see 9.2.2). It is set FALSE by the operation of the transmit timer state machine.

9.2.5.22 txTTL

This variable indicates the time remaining before information in the outgoing LLDPDU is no longer valid. The TTL field in the Time To Live TLV is set to txTTL during LLDPDU construction (see 9.1.2). The value of txTTL depends on the following:

a) During normal operation, txTTL is set to whichever is the smaller of the values represented by Equation (1) and Equation (2):

$$(\text{msgTxInterval} \times \text{msgTxHold}) + 1 \quad (1)$$

$$65535 \quad (2)$$

where msgTxInterval is defined in 9.2.5.7 and msgTxHold is defined in 9.2.5.6.

b) If adminStatus is transitioning to ‘disabled’ or portEnabled is transitioning to FALSE, txTTL shall be set to zero.

9.2.6 Per-Agent statistical counters

An instance of each of these statistical counters exists per LLDP agent; they are used to count significant events in the transmit and receive state machines and are made available to the management functions described in Clause 10 and Clause 11. All counters are maintained as unsigned integer values, four octets in length, and are initialized to zero only on system initialization (i.e., when the BEGIN system variable is set TRUE).

9.2.6.1 statsAgeoutsTotal

This counter provides a count of the times that a neighbor’s information has been deleted from the LLDP remote systems MIB because the rxInfoTTL timer associated with the neighbor’s MSAP has expired.

9.2.6.2 statsFramesDiscardedTotal

This counter provides a count of all LLDPDUs received and then discarded for any of the following reasons:

a) One or more of the three mandatory TLVs at the beginning of the LLDPDU is missing, out of order, or contains an out of range information string length.
b) There is insufficient space in the remote systems MIB to store the LLDPDU.

9.2.6.3 statsFramesInErrorsTotal

This counter provides a count of all LLDPDUs received with one or more detectable errors.

9.2.6.4 statsFramesInTotal

This counter provides a count of all LLDP frames received.

9.2.6.5 statsFramesOutTotal

This counter provides a count of all LLDP frames transmitted by this instance of the transmit state machine. The counter shall be maintained as an unsigned integer value, four octets in length.

9.2.6.6 statsTLVsDiscardedTotal

This counter provides a count of all TLVs received and then discarded for any reason.

9.2.6.7 statsTLVsUnrecognizedTotal

This counter provides a count of all TLVs received on the port that are not recognized by the LLDP local agent.

9.2.6.8 lldpduLengthErrors

A count of the number of LLPDU length restriction errors detected by the mibConstrInfoLLDPDU() procedure (9.2.7.2).

9.2.7 State machine procedures

9.2.7.1 dec (variable-name)

If the state machine variable *variable-name* has a non-zero value, this procedure decrements the value of the variable by 1.

9.2.7.2 mibConstrInfoLLDPDU()

The mibConstrInfoLLDPDU() procedure constructs an information LLDPDU, structured according to the specification in 8.2, the associated basic TLV formats defined in 8.4, plus any optional Organizationally Specific TLVs as specified and their associated individual organizationally defined formats (see 8.6). The information LLDPDU contains the following:

a) The set of mandatory TLVs as specified in 8.2, with the value of txTTL set in accordance with the definition in 9.2.5.22.
b) Additional optional TLVs, from the basic management set or from one or more organizationally specific sets, as allowed by LLDPDU length restrictions and as selected in the LLDP local system MIB by network management (see Table 8-1).

c) Optionally, an End Of LLDPDU TLV.

If the set of TLVs that is selected in the LLDP local system MIB by network management would result in an LLDPDU that violates LLDPDU length restrictions, then the variable lldpduLengthErrors (9.2.7.2) is incremented by 1, and an LLDPDU is sent containing the mandatory TLVs plus as many of the optional TLVs in the set as will fit in the remaining LLPDU length.

9.2.7.3 mibConstrShutdownLLDPDU()

The mibConstrShutdownLLDPDU() procedure constructs a shutdown LLDPDU according to the LLDPDU and the associated TLV formats specified in 8.2 and 8.5. The shutdown LLDPDU contains only the following items:

a) The Chassis ID and Port ID TLVs.
b) The Time To Live TLV with the TTL field set to zero.
c) Optionally, an End Of LLDPDU TLV.

9.2.7.4 mibDeleteObjects()

The mibDeleteObjects() procedure deletes all information in the LLDP remote systems MIB associated with a given MSAP identifier if an LLDPDU is received with an rxTTL value of zero (see 9.2.7.7.1) or the timing counter rxInfoTTL expires (see 9.2.2.1).

9.2.7.5 mibUpdateObjects()

The mibUpdateObjects() procedure updates the MIB objects corresponding to the TLVs contained in the received LLDPDU for the LLDP remote system if MIB space is available, as follows:

a) Compare the MSAP identifier in the current LLDPDU with the MSAP identifiers in the LLDP remote systems MIB:
    1) If a match is found, replace all current information associated with the MSAP identifier in the LLDP remote systems MIB with the information in the current LLDPDU.
    2) If no match is found, create a new MIB structure to receive information associated with the new MSAP identifier, and set these MIB objects to the values indicated in their respective TLVs.
b) Set the timing counter rxInfoTTL associated with the MSAP identifier to rxTTL.

If an incoming TLV is not recognized by the receiving LLDP agent, the TLV is stored in the LLDP remote systems MIB as follows:

c) If the TLV type value is in the range of the reserved TLV types in Table 8-1, the TLV can be from a later version of the basic management set and is stored according to the basic TLV format shown in Figure 8-2. These TLVs are indexed by their TLV type.
d) If the TLV type value is 127, the TLV is an Organizationally Specific TLV and is stored according to the basic format for Organizationally Specific TLVs shown in Figure 8-12. These TLVs are indexed by their OUI and organizationally defined TLV subtype.

9.2.7.6 rxInitializeLLDP()

The rxInitializeLLDP() procedure initializes the LLDP receive module. After the variable portEnabled is equal to TRUE, the LLDP receive module is initialized or re-initialized for frame reception. During this process, the local LLDP agent performs the following tasks:

a) The variable tooManyNeighbors is set to FALSE.
b) All information in the remote systems MIB associated with this LLDP agent is deleted.

9.2.7.7 rxProcessFrame()

The rxProcessFrame() procedure:

a) Validates the TLVs contained in the LLDPDU as defined in 9.2.7.7.1 through 9.2.7.7.3.
b) Determines whether or not a MIB update may be required as defined in 9.2.7.7.4.
c) If sufficient space is not available, determines whether to discard the incoming LLDPDU from a new neighbor or to delete information from an existing neighbor that is already in the LLDP remote systems MIB, as defined in 9.2.7.7.5. The tooManyNeighborsTimer and the tooManyNeighbors flag variable are both set during this process.

NOTE—The flag variable tooManyNeighbors is automatically reset when the tooManyNeighborsTimer expires.

9.2.7.7.1 LLDPDU validation

The receive module processes each incoming LLDPDU as it is received. The statsFramesInTotal counter for the port is incremented and the LLDPDU is checked to verify the presence of the three mandatory TLVs at the beginning of the LLDPDU as defined in 8.2.

a) The first TLV is extracted:
   1) If the extracted TLV type value does not equal 1, the TLV is not a Chassis ID TLV:
      i) The LLDPDU is discarded.
      ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
      iii) The variable badFrame is set to TRUE.
      iv) The procedure rxProcessFrame**()** is terminated.
   2) If the extracted TLV type value equals 1, and the chassis ID TLV information string length is not within the range $2 \leq n \leq 256$:
      i) The LLDPDU is discarded.
      ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
      iii) The variable badFrame is set to TRUE.
      iv) The procedure rxProcessFrame() is terminated.
   3) If the extracted TLV type value equals 1, and the chassis ID TLV information string length is within the range $2 \leq n \leq 256$, the chassis ID value is extracted to become the first part of the MSAP identifier.
b) The second TLV is extracted:
   1) If the extracted TLV type value does not equal 2, the TLV is not a Port ID TLV:
      i) The LLDPDU is discarded.
      ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
      iii) The variable badFrame is set to TRUE.
      iv) The procedure rxProcessFrame() is terminated.
   2) If the extracted TLV type value equals 2, and the port ID TLV information string length is not within the range $2 \leq n \leq 256$:
      i) The LLDPDU is discarded.
      ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
      iii) The variable badFrame is set to TRUE.

iv) The procedure rxProcessFrame() is terminated.

3) If the extracted TLV type value equals 2, and the port ID TLV information string length is within the range $2 \leq n \leq 256$, the port ID value is extracted and appended to the chassis ID value to complete construction of the MSAP identifier.

c) The third TLV is extracted:

1) If the extracted TLV type value does not equal 3, the TLV is not a Time To Live TLV.

i) The LLDPDU is discarded.

ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.

iii) The variable badFrame is set to TRUE.

iv) The procedure rxProcessFrame() is terminated.

2) If the extracted TLV type value equals 3, and the Time To Live TLV information string length is less than 2:

i) The LLDPDU is discarded.

ii) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.

iii) The variable badFrame is set to TRUE.

iv) The procedure rxProcessFrame() is terminated.

3) If the extracted TLV type value equals 3, and the Time To Live TLV information string length is greater than or equal to 2, the first two octets of the TLV information string is extracted and rxTTL is set to this value:

i) If rxTTL equals zero, a shutdown frame has been received. The MSAP identifier and rxTTL are passed up to the LLDP MIB manager, and further LLDPDU validation is terminated.

ii) If rxTTL is non-zero, LLDPDU validation continues and the remaining TLVs are validated.

Each of the remaining TLV information elements are decoded in succession as required by their particular TLV format definitions:

d) If the TLV type value equals 0, the TLV is the End Of LLDPDU TLV. The MSAP identifier, rxTTL, and all validated TLVs are passed to the LLDP management entity for LLDP remote systems MIB updating.

e) If ($0 <$ TLV_type_value ≤ 8) the TLV is a member of the basic management set and is validated according to the general rules for all TLVs defined in 9.2.7.7.2 as well as any specific rules defined for the particular TLVs defined in 8.5.

f) If TLV_type_value is in the range of reserved TLV types in Table 8-1, the TLV is unrecognized and may be a basic TLV from a later LLDP version. The statsTLVsUnrecognizedTotal counter is incremented, and the TLV is assumed to be validated.

g) If the TLV type value is 127, the TLV is an Organizationally Specific TLV:

1) If the TLV's OUI and organizationally defined subtype are recognized, the TLV is validated according to the general rules for all TLVs defined in 9.2.7.7.2 as well as the general rules for Organizationally Specific TLVs defined in 9.2.7.7.3 plus any specific rules defined for the particular TLV.

2) If the TLV's OUI and/or organizationally defined subtype are not recognized, the statsTLVsUnrecognizedTotal counter is incremented, and the TLV is assumed to be validated.

h) If the end of the LLDPDU has been reached, the MSAP identifier, rxTTL, and all validated TLVs are passed to the LLDP management entity for LLDP remote systems MIB updating.

9.2.7.7.2 General validation rules for all TLVs

The value in the TLV information string length field is the value used to validate the TLV and to indicate the location of the next TLV in the LLDPDU.

All TLVs are subject to the general validation rules listed in this subclause as well as any specific usage rules defined for the particular TLV (for example, see systems capabilities TLV usage rules in 8.5.8.3):

a) If the LLDPDU contains more than one Chassis ID TLV, Port ID TLV, or Time To Live TLV:
   1) The LLDPDU is discarded.
   2) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
   3) The variable badFrame is set to TRUE.
   4) The procedure rxProcessFrame() is terminated.
b) If the TLV information string length value is not greater than or equal to the sum of the lengths of all fields contained in the TLV information string:
   1) The LLDPDU is discarded.
   2) The statsFramesDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
   3) The variable badFrame is set to TRUE.
   4) The procedure rxProcessFrame() is terminated.
c) If any TLV contains an error condition specified for that particular TLV:
   1) The TLV is discarded.
   2) The statsTLVsDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
   3) The location of the next TLV is based on the TLV information string length value of the current TLV.
d) With the exception of the TTL TLV for which specific rules apply (see 9.2.7.7.1), if the length of any field of a TLV is outside the length range specified for that particular TLV (for example, the TLV information string length is greater than the maximum permitted length for the TLV information string as specified for that TLV):
   1) The TLV is discarded.
   2) The statsTLVsDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
   3) The location of the next TLV is based on the TLV information string length value of the current TLV.
e) If any TLV extends past the physical end of the frame:
   1) The TLV is discarded.
   2) The statsTLVsDiscardedTotal and statsFramesInErrorsTotal counters are both incremented.
f) Any information following an End Of LLDPDU TLV is ignored.

NOTE—Usage rules for individual TLVs allow some TLVs to appear more than once in an LLDPDU. Duplicate TLVs result in any one of the values being placed in the MIB, can cause the discard stats to increment, and can cause the change marker for the MIB entry to change if any of the TLV copies change the value even if the value finally recorded is unchanged. The only thing guaranteed is that the MIB value is set to one (unspecified) of the TLV values, and if that value is different to what was previously in the MIB then the change marker is set.

9.2.7.7.3 General validation rules for all Organizationally Specific TLVs

If an Organizationally Specific TLV is not recognized by the receiving LLDP agent, the content of the TLV can be ignored but the TLV is stored in the LLDP remote systems MIB for possible retrieval by network management using the basic Organizationally Specific TLV format (see 8.6) and the StatsTLVsUnrecognizedTotal counter is incremented. If more than one unrecognized Organizationally Specific TLV is received with the same OUI and organizationally defined subtype, but with identifiable differences in the organizationally defined information strings, all copies are assigned a temporary identification index and stored.

9.2.7.7.4 TLV/MIB object value comparison

The LLDP MIB manager is responsible for updating and maintaining the LLDP remote systems MIB with information received by LLDP agents. It uses the rxChanges variable to signal when any change has occurred in the LLDP remote systems MIB due to new or different information in the TLVs from an incoming LLDPDU.

The TLV/MIB object value comparison function is associated with the following actions:

Compare the MSAP identifier in the current LLDPDU with the MSAP identifiers in the LLDP remote systems MIB:

a) If a match is found, but no difference exists between the information in the TLVs just received and the information in the LLDP remote systems MIB, set the control variable rxChanges to FALSE, set the timing counter rxInfoTTL associated with the MSAP identifier to rxTTL, and wait for the next LLDPDU.
b) If a match is not found, or if a match is found and any differences exist between the information in the TLVs just received and the information in the LLDP remote systems MIB, check to determine if sufficient space exists in the LLDP remote systems MIB to accommodate the current LLDPDU:
    1) If sufficient space exists, set the control variable rxChanges to TRUE and set the flag variable tooManyNeighbors to FALSE.
    2) If sufficient space does not exist, perform the “Too many neighbors” process defined in 9.2.7.7.5.

9.2.7.7.5 Too many neighbors

When there is insufficient space in the LLDP remote systems MIB to store information from a new neighbor something needs to be discarded. Either the received LLDPDU is discarded or existing information within the current remote systems MIB is discarded in order to make space for the new information received in the LLDPDU:

a) Set the flag variable tooManyNeighbors to TRUE.
b) If the information selected to be discarded is the information in the current LLDPDU:
    1) Set the tooManyNeighborsTimer as specified in Equation (3).

$$\text{tooManyNeighborsTimer} = \max(\text{tooManyNeighborsTimer}, \text{rxTTL}) \quad (3)$$

    where rxTTL is the TTL value in the current LLDPDU
    2) Discard the current LLDPDU and increment the statsFramesDiscardedTotal counter.
    3) Wait for the next LLDPDU.
c) If the information selected to be discarded is currently in the LLDP remote systems MIB:
    1) Delete all information associated with the selected neighbor’s MSAP identifier from the LLDP remote systems MIB.
    2) Set the tooManyNeighborsTimer as specified in Equation (4).

$$\text{tooManyNeighborsTimer} = \max(\text{tooManyNeighborsTimer}, \text{rxInfoTTL}) \quad (4)$$

    where rxInfoTTL = the selected neighbor’s TTL value
    3) If sufficient space exists to store the information received in the current LLDPDU in the LLDP remote systems MIB, set control variable rxChanges to TRUE.
    4) If sufficient space still does not exist to store the information received in the current LLDPDU in the LLDP remote systems MIB, perform steps 1–3 again.

The variable tooManyNeighbors is automatically set to FALSE whenever the tooManyNeighborsTimer expires.

NOTE—The tooManyNeighborsTimer is not precise, as it is only cleared (and the tooManyNeighbors variable set FALSE) by timer expiration, and not by the removal of the too many neighbors condition.

9.2.7.8 somethingChangedLocal()

This procedure is called to indicate that the status/value of one or more of the selected objects in the LLDP local system MIB has changed, and that there is therefore a need to transmit new information. It is used to notify the managers of the PTOPO and other optional MIBs that something has changed in the LLDP local systems MIB.

This procedure also sets the value of the variable localChange (see 9.2.5.3) to signal the need for the LLDP agent(s) to transmit the new information, and copies the value of the variable txFastInit (see 9.2.5.19) to the variable txFast (see 9.2.5.18) to ensure that the new values will be seen by the neighbors on the port.

It is assumed that the system provides the ability for any processes that need to receive such indications to register with this procedure so that appropriate signals can be received by those processes when the procedure is called.

9.2.7.9 somethingChangedRemote()

This procedure is called after all the information associated with the particular MSAP identifier has been updated in the LLDP remote systems MIB. The procedure call serves as an indication to the managers of the PTOPO and other optional MIBs that one or more of the following conditions is true:

a) That the status/value of one or more objects in the LLDP remote systems MIB associated with the particular MSAP identifier has changed.
b) That an incoming LLDPDU contained an MSAP identifier requiring creation of a new MIB structure in the LLDP remote systems MIB to receive the information in that LLDPDU.
c) That information in the LLDP remote systems MIB associated with the MSAP identifier has been deleted.

It is assumed that the system provides the ability for any processes that need to receive such indications to register with this procedure so that appropriate signals can be received by those processes when the procedure is called.

9.2.7.10 txAddCredit()

This procedure increments the value of the txCredit variable (9.2.5.16) by one if the current value of the txCredit variable is less than the value of txCreditMax (9.2.5.17).

9.2.7.11 txFrame()

The txFrame() procedure makes use of the services of LLC to transmit the LLDPDU constructed by either the mibConstrInfoLLDPDU() or mibConstrShutdownLLDPDU() procedures (9.2.7.2, 9.2.7.3). The destination MAC address used is the MAC address associated with the instance of the LLDP Agent (see 7.1 and 9.2.5.4). The source MAC address used is the individual MAC address of the transmitting port. The EtherType used is the LLDP EtherType identified in Table 7-3.

After the frame has been transmitted, the statsFramesOutTotal counter (9.2.6.5) is incremented.

9.2.7.12 txInitializeLLDP()

The txInitializeLLDP() procedure initializes the LLDP transmit module. It performs the following tasks:

a) If applicable, either the non-volatile configuration for the LLDP local system MIB are retrieved or the appropriate default values are assigned to all LLDP configuration variables.
b) The internal (implementation specific) data structures are initialized with appropriate local physical topology information.
c) System default values are set for the following timing parameters:
   1) reinitDelay (9.2.5.10)
   2) msgTxHold (9.2.5.6)
   3) msgTxInterval (9.2.5.7)
   4) msgFastTx (9.2.5.5)

NOTE—It is recommended to introduce a degree of stagger into starting the LLDP local agent initialization in multi-port implementations so that the timing of LLDP frame transmission cycles can be distributed among system ports, and distributed among supported MAC addresses on each port. The method of accomplishing the stagger is an implementation issue and is beyond the scope of this standard.

9.2.8 Transmit state machine

The transmit state machine for an individual port and destination MAC address (see 7.1) shall implement the function specified by the state diagram contained in Figure 9-1 and the attendant definitions contained in 9.2.3 through 9.2.6.8.

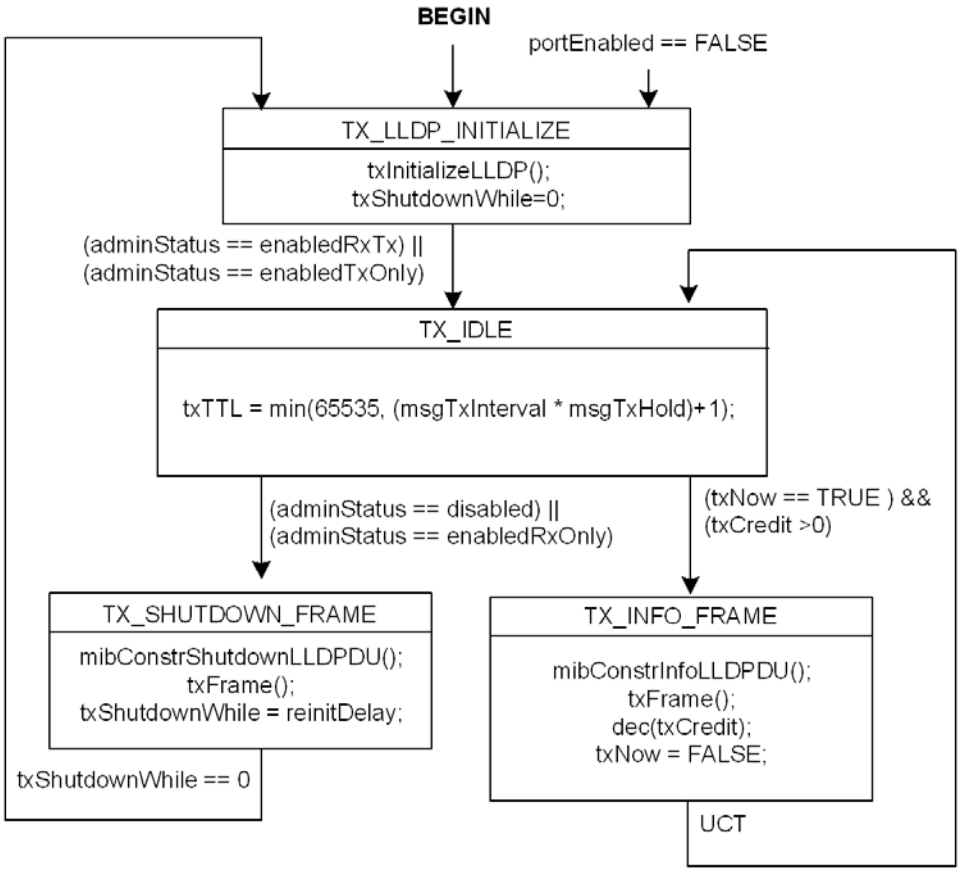

Figure 9-1—Transmit state machine

9.2.9 Receive state machine

The receive state machine for an individual port and destination MAC address (see 7.1) shall implement the function specified by the state diagram contained in Figure 9-2 and the attendant definitions contained in 9.2.2 through 9.2.6.8.

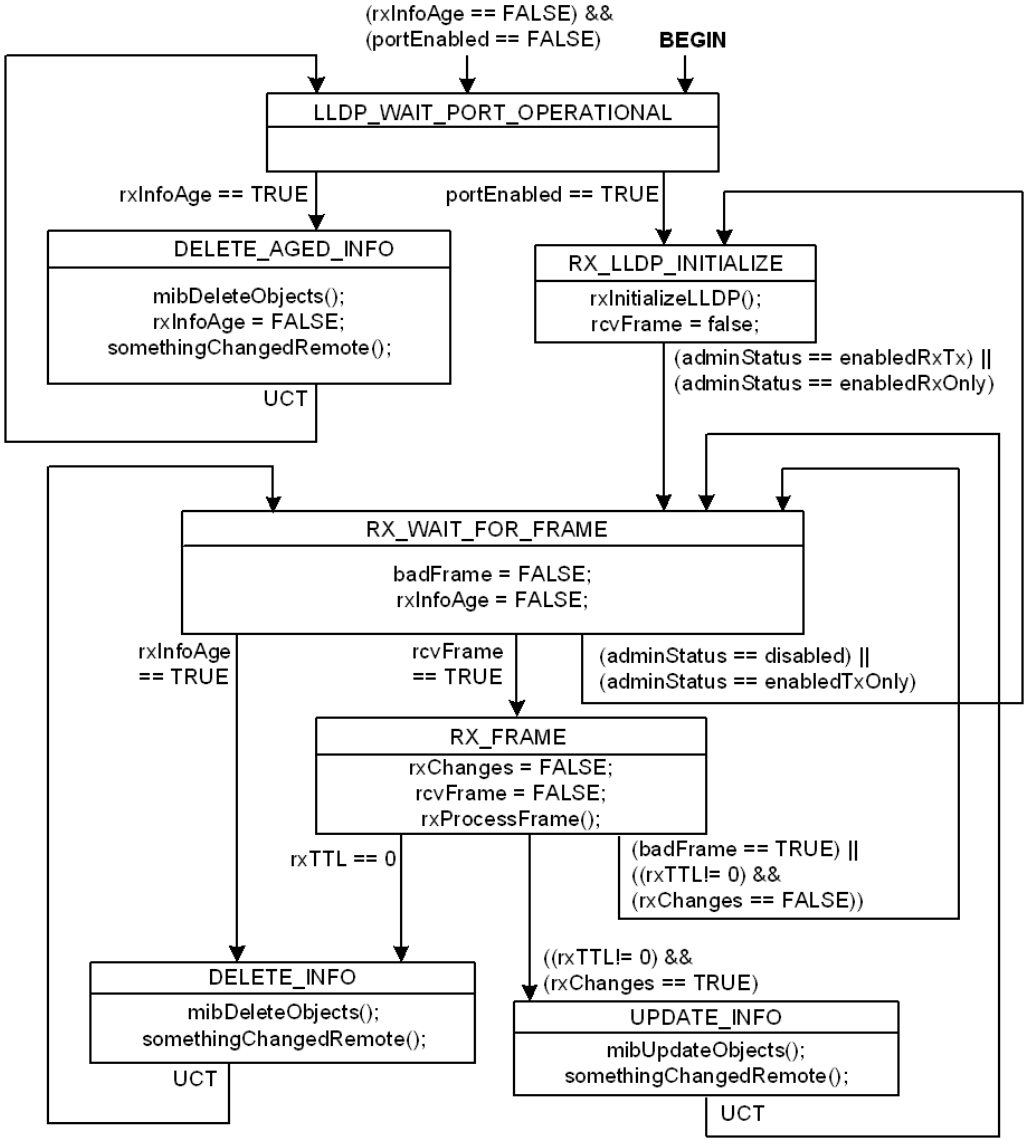

Figure 9-2—Receive state machine

9.2.10 Transmit timer state machine

The transmit timer state machine for an individual port and destination MAC address (see 7.1) shall implement the function specified by the state diagram contained in Figure 9-3 and the attendant definitions contained in 9.2.3 through 9.2.6.8.

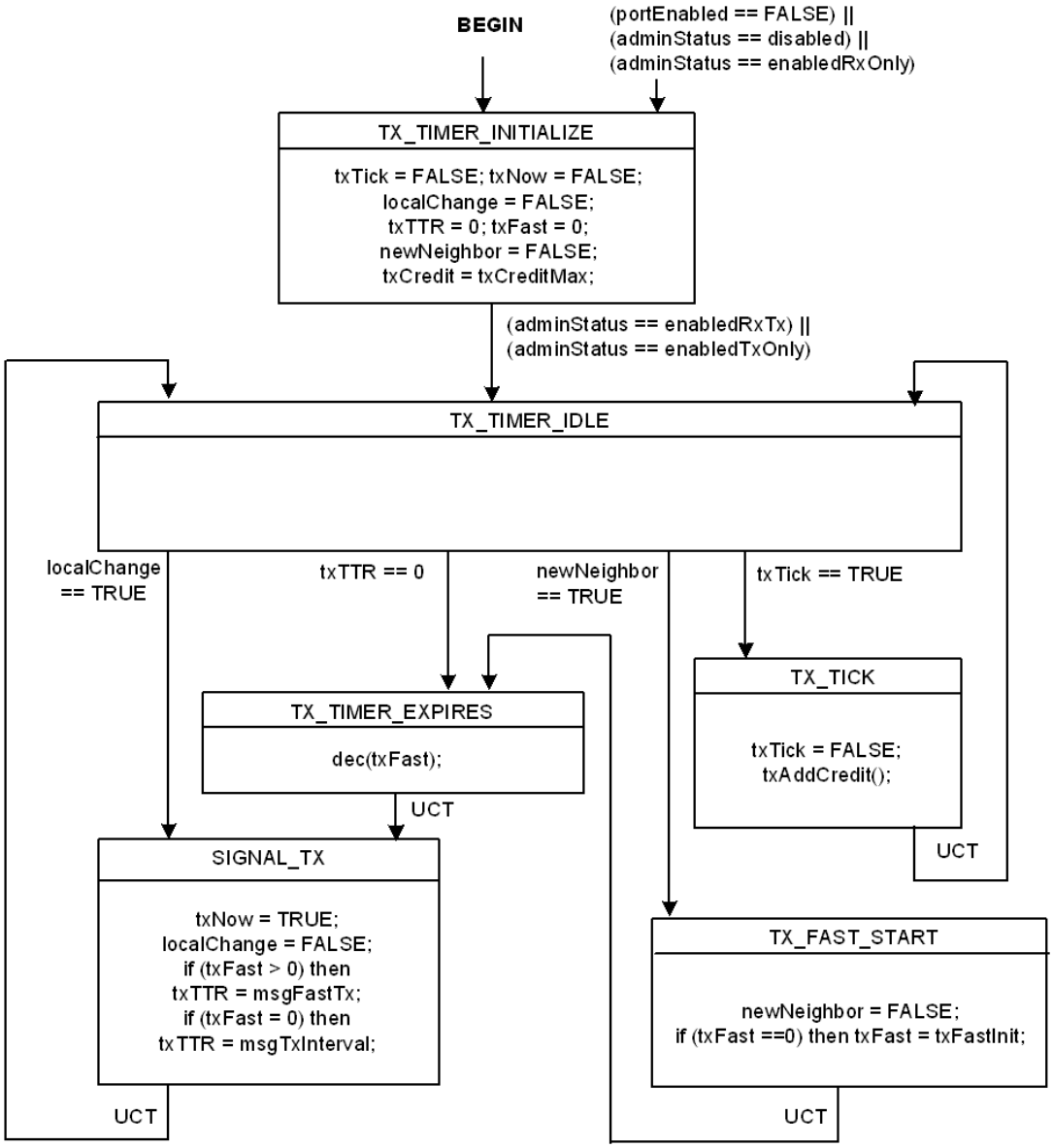

Figure 9-3—Transmit timer state machine

10. LLDP management

This clause defines the set of managed objects and their functionality that allow administrative configuration and monitoring of LLDP operation.

10.1 Data storage and retrieval

LLDP agents need to have a place to store both information about the local system and information they have received about remote systems. Data storage and retrieval capability to support the functionality defined in Clause 7, Clause 8, Clause 9, and Clause 10 shall be provided. No particular implementation is implied.

10.2 The LLDP management entity's responsibilities

The LLDP management entity has the following responsibilities:

a) Starting the transmit and receive state machines.
b) Providing a means for the network manager to select TLVs to be included in outgoing LLDPDUs.
c) Extracting the necessary information from the LLDP local system MIB and constructing the individual TLVs selected for insertion into the LLDPDU.
d) Prepending appropriate addressing and LLDP EtherType to the LLDPDU before submitting the MA_UNITDATA.request for frame transmission.
e) Updating the LLDP remote systems MIB and monitoring for MIB object adds, deletions and value changes.
f) Maintaining operational statistics.

10.2.1 Protocol initialization management

Protocol initialization consists of the following:

a) Retrieving non-volatile configuration values for the LLDP local system MIB.
b) Assigning default or management assigned values to LLDP configuration variables.
c) Loading physical topology information into their associated LLDP local system MIB objects.

10.2.2 TLV selection management

TLV selection management consists of providing the network manager with the means to select which specific TLVs are enabled for inclusion in an LLDPDU. The following LLDP variables cross reference to LLDP local systems configuration MIB tables indicating which specific TLVs are enabled for the particular port(s) on the system. The specific port(s) through which each TLV is enabled for transmission may be set (or reset) by the network manager.

a) **mibBasicTLVsTxEnable:** This variable lists the single-instance-use basic management TLVs, each with a bit map indicating the system ports through which the referenced TLV is enabled for transmission.
b) **mibMgmtAddrInstanceTxEnable:** This variable lists the different management addresses (by subtype) that are defined for the system, each with a bit map indicating the system ports through which the particular management address/subtype TLV is enabled for transmission.

NOTE—Implementers of new TLVs should be aware that provision needs to be made to allow similar network management selection of new TLVs and that doing so requires linkage to the LLDP MIB, regardless of whether or not the TLV is part of the basic management set or part of an Organizationally Specific TLV set.

10.2.3 Transmission management

Transmission management consists of the following:

a) Determining when a new transmission cycle is required, as signaled by the somethingChangedLocal() procedure (see 9.2.7.8).
b) Extracting the appropriate LLDP local system MIB information for the three mandatory TLVs and inserting them at the beginning of the LLDPDU in proper format order.
c) Extracting the appropriate LLDP local system MIB information for the selected optional TLVs and inserting them into the LLDPDU.
d) Optionally, appending an End Of LLDPDU TLV after the last optional TLV in the LLDPDU.
e) Maintaining length control during LLDPDU construction to ensure that the TLVs do not exceed the maximum length allowed for the LLDPDU.
f) Submitting the frame for transmission.

10.2.4 Reception management

Reception management consists of the following:

a) Receiving and parsing the incoming LLDPDUs.
b) Validating, and checking the types of the first three TLVs to ensure that they are the required type and in LLDPDU format order.
c) Validating optional TLVs and extracting information values.
d) Checking whether or not the current LLDPDU represents a new MSAP identifier.
e) Monitoring whether or not there is sufficient space available in the LLDP remote systems MIB for all active neighbors.
f) Arbitrating which information is to be discarded in cases where insufficient space is available in the LLDP remote systems MIB for all active neighbors.
g) Checking whether or not the received information represents a status or value change to the existing MIB object.
h) Updating remote systems MIB objects as necessary.

10.2.5 Performance management

Performance management consists of the following:

a) Monitoring the LLDP local system MIB update activities for status or value changes to selected MIB objects and:
    1) Calling the somethingChangedLocal() procedure (see 9.2.7.8) whenever a status/value change occurs.
    2) Initiating a new transmit cycle if txCredit > 0.
b) Monitoring the txTTR timer for countdown expiration and initiating a new transmit cycle if txCredit > 0.
c) Monitoring the rxInfoTTL timer for countdown expiration and:
    1) Deleting the associated objects from the LLDP remote systems MIB whenever the timing counter expires.
    2) Performing the procedure somethingChangedRemote() associated with the MSAP identifier.
d) Monitoring rxTTL in the incoming LLDPDUs for shutdown indication and:
    1) Deleting the associated objects from the LLDP remote systems MIB whenever rxTTL = 0.
    2) Performing the procedure somethingChangedRemote() associated with the MSAP identifier.

e) Monitoring the "tooManyNeighbors" variable to determine whether an existing neighbor's information needs to be deleted and:
   1) Deleting the objects associated with a selected MSAP identifier from the LLDP remote systems MIB whenever existing information is selected to be deleted.
   2) Performing the procedure somethingChangedRemote() associated with the MSAP identifier.

f) Monitoring the tooManyNeighborsTimer to determine when there is not sufficient space available to accommodate information from all active neighbors and setting the variable tooManyNeighbors associated with the port to FALSE when the tooManyNeighborsTimer expires.

g) Notifying the managers of the PTOPO MIB and other optional MIB modules (see Figure 11-1) whenever the procedures somethingChangedLocal() or somethingChangedRemote() are performed.

h) Maintaining operational statistics regarding the LLDP frames sent and received.

10.3 Managed objects

Managed objects model the semantics of management operations. Operations upon a managed object supply information concerning, or facilitate control over, the process or entity associated with that managed object.

Management of LLDP is described in terms of the managed resources that are associated with individual TLVs and that support frame transmission and reception.

10.4 Data types

This subclause specifies the semantics of operations independent of their encoding in management protocol. The data types of the parameters of operations are defined for that specification.

The following data types are used:

a) Boolean.
b) Enumerated, for a collection of named values.
c) Unsigned, for all parameters specified as "number of" some quantity.
d) MAC address.
e) Time interval, an Unsigned value representing a positive integral number of seconds for all time out parameters.
f) Counter, for all parameters specified as a "count" of some quantity (all counters increment and wrap with a modulus of 2 to the power 32).

10.5 LLDP variables

LLDP managed objects fall into the following categories:

a) Status/control variables necessary for operation of the protocol.
b) Variables that accumulate operational statistics.
c) Variables that are required by the particular TLVs.

10.5.1 LLDP operational status and control

a) Global parameters and variables:
   1) **adminStatus:** The authority that controls whether or not a local LLDP agent is enabled for transmit and receive, transmit only, or receive only; or is disabled (9.2.5.1).

b) Transmit state machine parameters and variables:
   1) **msgTxHold:** A multiplier on msgTxInterval used to compute the TTL value of txTTL (9.2.5.6, 9.2.5.22).
   2) **msgTxInterval:** The interval between successive transmit cycles in normal transmission mode (9.2.5.7).
   3) **reinitDelay:** The delay after adminStatus becomes 'disabled' before re-initialization is attempted (9.2.5.10).
   4) **txCreditMax:** The maximum value of transmission credit (9.2.5.17).
   5) **txFastInit:** The interval between successive LLDPDU transmissions in fast transmission mode (9.2.5.19).
c) Receive state machine parameters and variables:
   1) **remoteChanges:** An indication that the status/value of one or more selected objects in the LLDP remote systems MIB has changed or that all objects associated with a particular MSAP identifier have been deleted (9.2.5.11).
   2) **tooManyNeighbors:** An indication that there is insufficient space in the LLDP remote systems MIB to store information from all active neighbors (9.2.5.15).

10.5.2 LLDP operational statistics counters

The following counters provide operational statistics on a per port basis:

a) Transmission counters:
   1) **statsFramesOutTotal**: A count of all LLDP frames transmitted through the port (9.2.6.5).
   2) **statsLengthErrorsTotal:** A count of all LLDP length errors detected when constructing LLPDU frames for transmission through the port (9.2.7.2).
b) Reception counters:
   1) **statsAgeoutsTotal:** A count of the times that a neighbor's information is deleted from the LLDP remote systems MIB because of rxInfoTTL timer expiration (9.2.6.1).
   2) **statsFramesDiscardedTotal:** A count of all LLDPDUs received and then discarded (9.2.6.2).
   3) **statsFramesInErrorsTotal:** A count of all LLDPDUs received at the port with one or more detectable errors 9.2.6.3).
   4) **statsFramesInTotal:** A count of all LLDP frames received at the port (9.2.6.4).
   5) **statsTlvsDiscardedTotal:** A count of all TLVs received at the port and discarded for any reason. (9.2.6.6)
   6) **statsTLVsUnrecognizedTotal:** This counter provides a count of all TLVs not recognized by the receiving LLDP local agent (9.2.6.7).

10.5.3 TLV required variables

Variables in this category are defined by the requirements of the particular TLVs. They are maintained with reference to the local system in the LLDP local system MIB; and with reference to the remote system, in the LLDP remote systems MIB. TLV variables are outputs from the local LLDP agent and inputs to the remote LLDP agent.

Variables that pertain to basic set TLVs are listed in the following subclauses.

10.5.3.1 Chassis ID TLV objects

a) **chassis ID subtype:** The type of identifier used for the chassis (see 8.5.2.2).
b) **chassis ID:** The identification assigned to the chassis containing the port (see 8.5.2.3).

10.5.3.2 Port ID TLV objects

a) **port ID subtype:** The type of identifier used for the port (see 8.5.3.2).
b) **port ID:** The identification assigned to the port (see 8.5.3.3).

10.5.3.3 Port description TLV object

a) **port description:** The port's description (see 8.5.5.2).

10.5.3.4 System name TLV object

a) **system name:** The system's assigned name (see 8.5.6.2).

10.5.3.5 System description TLV object

a) **system description:** The system's description (see 8.5.7.2).

10.5.3.6 System capabilities TLV objects

a) **system capabilities:** The primary capabilities of the system (see 8.5.8.1).
b) **enabled capabilities:** The system's enabled capabilities (see 8.5.8.2).

10.5.3.7 Management address TLV objects

a) **management address length:** The length of the management address (see 8.5.9.2).
b) **management address subtype:** The management address type (see 8.5.9.3).
c) **management address:** The management address (see 8.5.9.4).
d) **interface numbering subtype:** The interface type (see 8.5.9.5).
e) **interface number:** The interface number (see 8.5.9.6).
f) **OID length:** The object identifier length (see 8.5.9.7).
g) **OID:** The object identifier (see 8.5.9.8).

11. LLDP MIB definitions

This clause defines the LLDP basic MIB for use with SNMP in TCP/IP based internets. In particular, it defines objects for managing the operation of LLDP based on the specifications of Clause 7, Clause 8, Clause 9, and Clause 10.

11.1 Internet Standard Management Framework

LLDP MIBs are designed to operate in a manner consistent with the principles of the Internet Standard Management Framework, which describes the separation of a data modeling language (for example, SMIv2) from content-specific data models (for example, the LLDP remote systems MIB), and from messages and protocol operations used to manipulate the data (for example, SNMPv3).

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

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 clause specifies a MIB module that is compliant to the SMIv2, which is described in IETF RFC 2578 (STD 58) [B3], IETF RFC 2579 (STD 58) [B4], and IETF RFC 2580 (STD 58) [B5].

11.2 Structure of the LLDP MIB

The LLDP MIB consists of two types of MIB modules, the mandatory basic MIB defined in this clause and from zero to *n* optional organizationally specific MIB extensions such as the IEEE 802.1 MIB in Annex D of IEEE Std 802.1Q-2014 and the IEEE 802.3 MIB in Clause 79 of IEEE Std 802.3-2012.

Each MIB module is divided into two major sections as shown in Figure 11-1 to allow selective MIB support for the particular operating mode (transmit only, receive only, or both transmit and receive) being implemented.

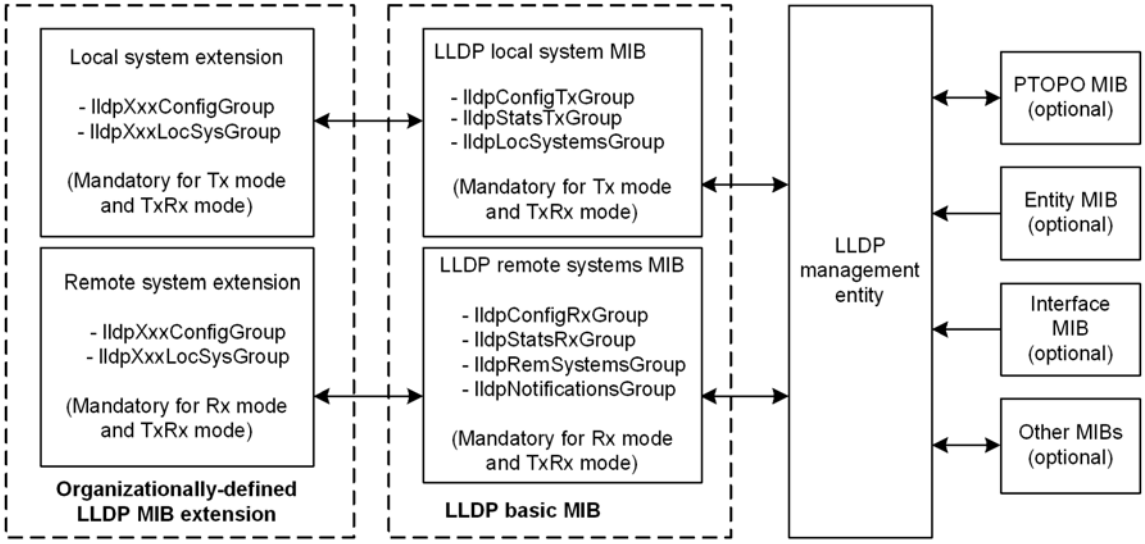

Figure 11-1—LLDP MIB block diagram

Table 11-1 summarizes the particular object groups that are required for each operating mode.The basic MIB shall comply with the MIB conformance section for the particular operating mode (Rx, Tx, or TxRx mode) being supported.

Table 11-1—MIB object groups and operating mode applicability

MIB group	Rx mode	Tx mode	TxRx mode
lldpV2ConfigGroup			
lldpV2ConfigRxGroup	M[a]	—	M
lldpV2ConfigTxGroup	—	M	M
lldpV2StatsRxGroup	M	—	M
lldpV2StatsTxGroup	—	M	M
lldpV2LocSysGroup	—	M	M
lldpV2RemSysGroup	M	—	M
lldpV2NotificationsGroup	M	—	M
ifGeneralInformationGroup	M	M	M

[a]M=Mandatory

Table 11-2 shows the structure of the MIB and the relationship of the MIB objects to the LLDP operational status/control variables, LLDP statistics variables, and TLV variables.

Table 11-2—LLDP MIB structure and object cross reference

MIB table	MIB object	LLDP reference
LLDP Configuration group		
	lldpV2MessageTxInterval	msgTxInterval, 9.2.5.7
	lldpV2MessageTxHoldMultiplier	msgTxHold, 9.2.5.6
	lldpV2ReinitDelay	reinitDelay, 9.2.5.10
	lldpV2NotificationInterval	msgTxInterval, 9.2.5.7
	lldpV2TxCreditMax	txCreditMax, 9.2.5.17
	lldpV2MessageFastTx	msgFastTx, 9.2.5.5
	lldpV2TxFastInit	txFastInit, 9.2.5.19

Table 11-2—LLDP MIB structure and object cross reference *(continued)*

MIB table	MIB object	LLDP reference
lldpV2PortConfigTableV2		
	lldpV2PortConfigIfIndexV2	(Table index)
	lldpV2PortConfigDestAddressIndexV2	(Table index)
	lldpV2PortConfigAdminStatusV2	adminStatus, 9.2.5.1
	lldpV2PortMessageTxInterval	msgTxInterval, 9.2.5.7
	lldpV2PortMessageTxHoldMultiplier	msgTxHold, 9.2.5.6
	lldpV2PortReinitDelay	reinitDelay, 9.2.5.10
	lldpV2PortNotificationInterval	msgTxInterval, 9.2.5.7
	lldpV2PortTxCreditMax	txCreditMax, 9.2.5.17
	lldpV2PortMessageFastTx	msgFastTx, 9.2.5.5
	lldpV2PortTxFastInit	txFastInit, 9.2.5.19
	lldpV2PortConfigNotificationEnableV2	—
	lldpV2PortConfigTLVsTxEnableV2	9.1.2.1
lldpV2DestAddressTable		
	lldpV2AddressTableIndex	(Table index)
	lldpV2DestMacAddress	(Table index)
lldpV2ManAddrConfigTxPortsTable		
	lldpV2ManAddrConfigIfIndex	(Table index)
	lldpV2ManAddrConfigDestAddressIndex	(Table index)
	lldpV2ManAddrConfigLocManAddrSubtype	8.5.9.3 (Table index)
	lldpV2ManAddrConfigLocManAddr	8.5.9.4 (Table index)
	lldpV2ManAddrConfigTxEnable	9.1.2.1
	lldpV2ManAddrConfigRowStatus	—
LLDP Statistics group		
	lldpV2StatsRemTablesLastChangeTime	—
	lldpV2StatsRemTablesInserts	—
	lldpV2StatsRemTablesDeletes	—
	lldpV2StatsRemTablesDrops	—
	lldpV2StatsRemTablesAgeouts	—

Table 11-2—LLDP MIB structure and object cross reference *(continued)*

MIB table	MIB object	LLDP reference
lldpV2StatsTxPortTable		
	lldpV2StatsTxIfIndex	(Table index)
	lldpV2StatsTxDestMACAddress	(Table index)
	lldpV2StatsTxPortFramesTotal	statsFramesOutTotal, 9.2.6.5
	lldpV2StatsTxLLDPDULengthErrors	lldpduLengthErrors, 9.2.6.8
lldpV2StatsRxPortTable		
	lldpV2StatsRxDestIfIndex	(Table index)
	lldpV2StatsRxDestMACAddress	(Table index)
	lldpV2StatsRxPortFramesDiscardedTotal	statsFramesDiscardedTotal, 9.2.6.2
	lldpV2StatsRxPortFramesErrors	statsFramesInErrorsTotal, 9.2.6.3
	lldpV2StatsRxPortFramesTotal	statsFramesInTotal, 9.2.6.4
	lldpV2StatsRxPortTLVsDiscardedTotal	statsTLVsDiscardedTotal, 9.2.6.6
	lldpV2StatsRxPortTLVsUnrecognizedTotal	statsTLVsUnrecognizedTotal, 9.2.6.7
	lldpV2StatsRxPortAgeoutsTotal	statsAgeoutsTotal, 9.2.6.1
Local System Data group		
	lldpV2LocChassisIdSubtype	chassis ID subtype, 8.5.2.2
	lldpV2LocChassisId	chassis ID, 8.5.2.3
	lldpV2LocSysName	system name, 8.5.6.2
	lldpV2LocSysDesc	system description, 8.5.7.2
	lldpV2LocSysCapSupported	system capabilities, 8.5.8.1
	lldpV2LocSysCapEnabled	enabled capabilities, 8.5.8.2
lldpV2LocPortTable		
	lldpV2LocPortIfIndex	(Table index)
	lldpV2LocPortIdSubtype	port ID subtype, 8.5.3.2
	lldpV2LocPortId	port ID, 8.5.3.3
	lldpV2LocPortDesc	port description, 8.5.5.2
lldpV2LocManAddrTable		
	lldpV2LocManAddrSubtype	management address subtype, 8.5.9.3 (Table index)
	lldpV2LocManAddr	management address, 8.5.9.4 (Table index)
	lldpV2LocManAddrLen	management address string length, 8.5.9.2
	lldpV2LocManAddrIfSubtype	interface numbering subtype, 8.5.9.5

Table 11-2—LLDP MIB structure and object cross reference *(continued)*

MIB table	MIB object	LLDP reference
	lldpV2LocManAddrIfId	interface number, 8.5.9.6
	lldpV2LocManAddrOID	object identifier, 8.5.9.8
Remote Systems Data group		
lldpV2RemTable		
	lldpV2RemTimeMark	(Table index)
	lldpV2RemLocalIfIndex	(Table index)
	lldpV2RemLocalDestMACAddress	(Table index)
	lldpV2RemIndex	(Table index)
	lldpV2RemChassisIdSubtype	chassis ID subtype, 8.5.2.2
	lldpV2RemChassisId	chassis ID, 8.5.2.3
	lldpV2RemPortIdSubtype	port ID sybtype, 8.5.3.2
	lldpV2RemPortId	port ID, 8.5.3.3
	lldpV2RemPortDesc	port description, 8.5.5.2
	lldpV2RemSysName	system name, 8.5.6.2
	lldpV2RemSysDesc	system description, 8.5.7.2
	lldpV2RemSysCapSupported	system capabilities, 8.5.8.1
	lldpV2RemSysCapEnabled	enabled capabilities, 8.5.8.2
	lldpV2RemRemoteChanges	remoteChanges, 9.2.5.11
	lldpV2RemTooManyNeighbors	tooManyNeighbors, 9.2.5.15
lldpV2RemManAddrTable		(Table index)
	lldpV2RemTimeMark	(Table index)
	lldpV2RemLocalIfIndex	(Table index)
	lldpV2RemLocalDestMACAddress	(Table index)
	lldpV2RemIndex	(Table index)
	lldpV2RemManAddrSubtype	management address subtype, 8.5.9.3 (Table index)
	lldpV2RemManAddr	management address, 8.5.9.4 (Table index)
	lldpV2RemManAddrIfSubtype	interface numbering subtype, 8.5.9.5
	lldpV2RemManAddrIfId	interface number, 8.5.9.6
	lldpV2RemManAddrOID	object identifier, 8.5.9.8

Table 11-2—LLDP MIB structure and object cross reference *(continued)*

MIB table	MIB object	LLDP reference
lldpV2RemUnknownTLVTable		
	lldpV2RemTimeMark	(Table index)
	lldpV2RemLocalIfIndex	(Table index)
	lldpV2RemLocalDestMACAddress	(Table index)
	lldpV2RemIndex	(Table index)
	lldpV2RemUnknownTLVType	LLDPDU validation, 9.2.7.7.1 (Table index)
	lldpV2RemUnknownTLVInfo	LLDPDU validation, 9.2.7.7.1
lldpV2RemOrgDefInfoTable		
	lldpV2RemTimeMark	(Table index)
	lldpV2RemLocalIfIndex	(Table index)
	lldpV2RemLocalDestMACAddress	(Table index)
	lldpV2RemIndex	(Table index)
	lldpV2RemOrgDefInfoOUI	organizationally unique identifier, 8.6.1.3 (Table index)
	lldpV2RemOrgDefInfoSubtype	organizationally defined subtype, 8.6.1.4 (Table index)
	lldpV2RemOrgDefInfoIndex	(Table index)
	lldpV2RemOrgDefInfo	organizationally defined information, 8.6.1.5
LLDP MIB Notifications		
	lldpV2RemTablesChange	

11.3 Relationship to other MIBs

This clause includes specifications for an LLDP Textual Conventions MIB module, an LLDP MIB module, and for IEEE 802.1 and IEEE 802.3 extension MIB modules that are compliant with the SMIv2 as defined in IETF RFC 2578 (STD 58) [B3], IETF RFC 2579 (STD 58) [B4], and IETF RFC 2580 (STD 58) [B5].

LLDP is designed to operate in conjunction with MIB modules defined by the IETF, IEEE 802, and others. The following subclauses discuss the relationship between LLDP and IETF MIB modules. LLDP agents automatically notify the managers of these MIB modules whenever there is a value or status change in an LLDP MIB object.

LLDP managed objects for the local system are stored in the LLDP local system MIB. Information received from a remote LLDP agent is stored in the local LLDP agent's LLDP remote system MIB.

NOTE—In this standard, managed objects for an LLDP MIB module are defined as they would be for an SNMP MIB. However, it is not required for LLDP implementations to support SNMP to store and retrieve system data. LLDP agents need to have a place to store both information about the local system and information they have received about remote systems. No particular implementation is implied.

11.3.1 Relationship to LLDP Version 1 MIB

The version 1 MIB module that was published in the initial 2005 publication of this standard has been superseded by the version 2 MIB module specified in 11.5.1, and support of the version 2 module is a requirement for conformance to the required or optional capabilities (Clause 5) in this revision of the standard. The version 2 MIB module reflects changes in indexation of the MIB objects that support the use of LLDP with multiple destination MAC addresses, as discussed in Clause 6.

In addition to the changes in indexation, the version 2 MIB module also supports changes to the management of the LLDP protocol that have resulted from changes to the LLDP protocol state machines. As these protocol changes affect only the timing and frequency of transmission of LLDPDUs, and not the content of the LLDPDUs, version 1 and version 2 implementations can successfully co-exist and interoperate.

From the perspective of a version 2 implementation, a version 1 neighbor behaves as a version 2 neighbor that is only using a single destination MAC address (the 01-80-C2-00-00-0E address specified for use in the IEEE 802.1AB-2005 version of the standard). From the perspective of a version 1 implementation, a version 2 neighbor that uses the 01-80-C2-00-00-0E destination MAC address behaves as a version 1 neighbor.

11.3.2 IETF Physical Topology MIB

The IETF Physical Topology (PTOPO) MIB (IETF RFC 2922 [B9]) allows a LLDP agent to expose learned physical topology information, using an IETF MIB. LLDP is intended to support the PTOPO MIB.

NOTE—The LLDP MIB module is a logical superset of the IETF PTOPO MIB; therefore, from a functional point of view, if the LLDP MIB module is implemented, it is likely not to be necessary to implement the PTOPO MIB defined in IETF RFC 2922 [B9]. However, for backward compatibility, this standard also supports the use of the PTOPO MIB.

11.3.3 IETF Entity MIB

The Entity MIB (IETF RFC 6933) allows the physical component inventory and hierarchy to be identified. Chassis IDs passed in the LLDPDU can identify entPhysicalTable entries. SNMP agents that implement the LLDP MIB should implement the entPhysicalAlias object from the Entity MIB version 2 or higher.

11.3.4 IETF Interfaces MIB

The Interfaces MIB (IETF RFC 2863) provides a standard mechanism for managing network ports. Port IDs passed in the LLDPDU can identify ifTable (or entPhysicalTable) entries. SNMP agents that implement the LLDP MIB shall also implement the ifTable and ifXTable for the ports that are represented in the Interfaces MIB.

11.4 Security considerations for LLDP base MIB module

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write.[10] 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.

[10]In IETF MIB definitions, the MAX-ACCESS clause defines the type of access that is allowed for particular data elements in the MIB. An explanation of the MAX-ACCESS mappings is given in section 7.3 of IETF RFC 2578 [B3].

a) Setting the following objects to incorrect values can result in an excessive number of LLDP packets being sent by the LLDP agent:
   1) lldpV2MessageTxInterval, lldpV2PortMessageTxInterval
   2) lldpV2TxCreditMax, lldpV2PortTxCreditMax
   3) lldpV2MessageFastTx, lldpV2PortMessageFastTx
   4) lldpV2TxFastInit, lldpV2PortTxFastInit
b) Setting the object, lldpV2MessageTxHoldMultiplier or lldpV2PortMessageTxHoldMultiplier, to incorrect values can cause the LLDP agent to transmit LLDPDUs with too-high TTL values, which affect the expiration time of objects grouped under lldpV2RemoteSystemsData identifier.
c) Setting the object, lldpV2ReinitDelay or lldpV2PortReinitDelay, to too low a value can cause the transmit state machine to attempt excessive re-initializations.
d) Setting incorrect bits in the object, lldpV2PortConfigTLVsTxEnableV2, can cause the LLDP agent to transmit LLDPDUs with an undesired optional TLV sequence.
e) Setting incorrect bits in the object, lldpV2ConfigManAddrPortsTxEnable, can cause the LLDP agent to advertise management addresses that were not meant to be disclosed and/or to omit addresses that were desired.
f) Setting the following objects to incorrect values can result in improper operation of the MIB notification process:
   1) llldpV2NotificationInterval
   2) llldpV2PortNotificationInterval
   3) lldpV2PortConfigNotificationEnableV2
g) Setting the object, lldpV2PortConfigAdminStatusV2, to the incorrect value can result in enabling a non-desired operational mode.

The following readable objects in this MIB module may be considered to be sensitive or vulnerable in some network environments:

h) Objects that are associated with the transmit mode
   1) lldpV2LocChassisIdSubtype
   2) lldpV2LocChassisId
   3) lldpV2LocPortIdSubtype
   4) lldpV2LocPortId
   5) lldpV2LocPortDesc
   6) lldpV2LocSysName
   7) lldpV2LocSysDesc
   8) lldpV2LocSysCapSupported
   9) lldpV2LocSysCapEnabled
   10) lldpV2LocManAddrLen
   11) lldpV2LocManAddrIfSubtype
   12) lldpV2LocManAddrIfId
   13) lldpV2LocManAddrOID
i) Objects that are associated with the receive mode
   1) lldpV2NotificationInterval
   2) lldpV2PortConfigNotificationEnable
   3) lldpV2RemChassisIdSubtype
   4) lldpV2RemChassisId
   5) lldpV2RemPortIdSubtyp
   6) lldpV2RemPortId
   7) lldpV2RemPortDesc
   8) lldpV2RemSysName
   9) lldpV2RemSysDesc
   10) lldpV2RemSysCapSupported
   11) lldpV2RemSysCapEnabled
   12) lldpV2RemManAddrIfSubtype

13) lldpV2RemManAddrIfId
14) lldpV2RemManAddrOID
15) lldpV2RemUnknownTLVInfo
16) lldpV2RemOrgDefInfo

This concern applies both to objects that describe the configuration of the local host, as well as for objects that describe information from the remote hosts, acquired via LLDP and displayed by the objects in this MIB module. 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.

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.

Implementers should consider the security features as provided by the SNMPv3 framework (see IETF RFC 3410, section 8), including full support for the SNMPv3 cryptographic mechanisms (for authentication and privacy).

Further, implementers should not deploy SNMP versions prior to SNMPv3. Instead**,** implementers should deploy SNMPv3 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.

11.5 LLDP MIB modules [11,12]

Two MIB modules are defined—a textual conventions MIB module (11.5.1) that contains the textual conventions used by the LLDP version 2 MIB module and by the version 2 extension MIB module in Annex D of IEEE Std 802.1Q-2014, and the LLDP version 2 MIB module itself (11.5.2). The textual conventions defined in the Textual-Convention MIB module are also available by extension MIB modules defined in other standards, such as the extension MIB modules defined in IEEE Std 802.1Q.

[11] *Copyright release for MIBs:* Users of this standard may freely reproduce the MIB contained in this subclause so that it can be used for its intended purpose.

[12] An ASCII version of this MIB module can be obtained by Web browser from the IEEE 802.1 Website at http://www.ieee802.org/1/pages/MIBS.html.

11.5.1 Definitions for the LLDP-V2-TC MIB module

In the following MIB module, should any discrepancy between the DESCRIPTION text and the corresponding definition in Clause 9 and Clause 10 occur, the definition in Clause 9 and Clause 10 shall take precedence.

```
LLDP-V2-TC-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    Unsigned32,
    org
        FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
        FROM SNMPv2-TC;

lldpV2TcMIB MODULE-IDENTITY
    LAST-UPDATED "201603110000Z" -- March 11, 2016
ORGANIZATION "IEEE 802.1 Working Group"
    CONTACT-INFO
        "  WG-URL: http://www.ieee802.org/1/
         WG-EMail: stds-802-1-L@ieee.org

          Contact: IEEE 802.1 Working Group Chair
           Postal: C/O IEEE 802.1 Working Group
                   IEEE Standards Association
                   445 Hoes Lane
                   Piscataway
                   NJ 08854
                   USA
          E-mail: stds-802-1-L@ieee.org"
    DESCRIPTION
            "Textual conventions used throughout the IEEE Std 802.1AB
            version 2 and later MIB modules.

            Unless otherwise indicated, the references in this
            MIB module are to IEEE 802.1AB-2016.

            The TCs in this MIB are taken from the original LLDP-MIB,
            LLDP-EXT-DOT1-MIB, and LLDP-EXT-DOT3-MIB published in
            IEEE Std 802-1D-2004, with the addition of TCs to support
            the management address table. They have been made available
            as a separate TC MIB module to facilitate referencing from
            other MIB modules.

            Copyright (C) IEEE (2016). This version of this MIB module
            is published as 11.5.1 of IEEE Std 802.1AB-2016;
            see the standard itself for full legal notices."

    REVISION "201603110000Z" -- March 11, 2016

    DESCRIPTION
            "Published as part of IEEE Std 802.1AB-2016.
            This revision updated referencees."

    REVISION "200906080000Z" -- June 08, 2009


    DESCRIPTION
```

```
            "Published as part of IEEE Std 802.1AB-2009 revision."

   ::= { org ieee(111) standards-association-numbers-series-standards(2)
         lan-man-stds(802) ieee802dot1(1) 1 12 }

--
-- Definition of the root OID arc for IEEE 802.1 MIBs
--

ieee802dot1mibs OBJECT IDENTIFIER
    ::= { org ieee(111) standards-association-numbers-series-standards(2)
          lan-man-stds(802) ieee802dot1(1) 1 }

--
-- ************************************************************
--
-- Textual Conventions
--
-- ************************************************************

LldpV2ChassisIdSubtype ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
            "This TC describes the source of a chassis identifier.

            The enumeration 'chassisComponent(1)' represents a chassis
            identifier based on the value of entPhysicalAlias object
            (defined in IETF RFC 6933) for a chassis component (i.e.,
            an entPhysicalClass value of 'chassis(3)').

            The enumeration 'interfaceAlias(2)' represents a chassis
            identifier based on the value of ifAlias object (defined in
            IETF RFC 2863) for an interface on the containing chassis.

            The enumeration 'portComponent(3)' represents a chassis
            identifier based on the value of entPhysicalAlias object
            (defined in IETF RFC 6933) for a port or backplane
            component (i.e., entPhysicalClass value of 'port(10)' or
            'backplane(4)'), within the containing chassis.

            The enumeration 'macAddress(4)' represents a chassis
            identifier based on the value of a unicast source address
            (encoded in network byte order and IEEE 802.3 canonical bit
            order), of a port on the containing chassis as defined in
            IEEE Std 802.

            The enumeration 'networkAddress(5)' represents a chassis
            identifier based on a network address, associated with
            a particular chassis. The encoded address is actually
            composed of two fields. The first field is a single octet,
            representing the IANA AddressFamilyNumbers value for the
            specific address type, and the second field is the network
            address value.

            The enumeration 'interfaceName(6)' represents a chassis
            identifier based on the value of ifName object (defined in
            IETF RFC 2863) for an interface on the containing chassis.

            The enumeration 'local(7)' represents a chassis identifier
```

```
            based on a locally defined value."
        SYNTAX  INTEGER {
                chassisComponent(1),
                interfaceAlias(2),
                portComponent(3),
                macAddress(4),
                networkAddress(5),
                interfaceName(6),
                local(7)
        }

    LldpV2ChassisId ::= TEXTUAL-CONVENTION
        DISPLAY-HINT "1x:"
        STATUS      current
        DESCRIPTION
                "This TC describes the format of a chassis identifier string.
                Objects of this type are always used with an associated
                LldpChassisIdSubtype object, which identifies the format of
                the particular LldpChassisId object instance.

                If the associated LldpChassisIdSubtype object has a value of
                'chassisComponent(1)', then the octet string identifies
                a particular instance of the entPhysicalAlias object
                (defined in IETF RFC 6933) for a chassis component (i.e.,
                an entPhysicalClass value of 'chassis(3)').

                If the associated LldpChassisIdSubtype object has a value
                of 'interfaceAlias(2)', then the octet string identifies
                a particular instance of the ifAlias object (defined in
                IETF RFC 2863) for an interface on the containing chassis.
                If the particular ifAlias object does not contain any values,
                another chassis identifier type should be used.

                If the associated LldpChassisIdSubtype object has a value
                of 'portComponent(3)', then the octet string identifies a
                particular instance of the entPhysicalAlias object (defined
                in IETF RFC 6933) for a port or backplane component within
                the containing chassis.

                If the associated LldpChassisIdSubtype object has a value of
                'macAddress(4)', then this string identifies a particular
                unicast source address (encoded in network byte order and
                IEEE 802.3 canonical bit order), of a port on the containing
                chassis as defined in IEEE Std 802.

                If the associated LldpChassisIdSubtype object has a value of
                'networkAddress(5)', then this string identifies a particular
                network address, encoded in network byte order, associated
                with one or more ports on the containing chassis. The first
                octet contains the IANA Address Family Numbers enumeration
                value for the specific address type, and octets 2 through
                N contain the network address value in network byte order.

                If the associated LldpChassisIdSubtype object has a value
                of 'interfaceName(6)', then the octet string identifies
                a particular instance of the ifName object (defined in
                IETF RFC 2863) for an interface on the containing chassis.
                If the particular ifName object does not contain any values,
                another chassis identifier type should be used.
```

```
            If the associated LldpChassisIdSubtype object has a value of
            'local(7)', then this string identifies a locally assigned
            Chassis ID."
    SYNTAX      OCTET STRING (SIZE (1..255))

LldpV2PortIdSubtype ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
            "This TC describes the source of a particular type of port
            identifier used in the LLDP MIB.

            The enumeration 'interfaceAlias(1)' represents a port
            identifier based on the ifAlias MIB object, defined in IETF
            RFC 2863.

            The enumeration 'portComponent(2)' represents a port
            identifier based on the value of entPhysicalAlias (defined in
            IETF RFC 6933) for a port component (i.e., entPhysicalClass
            value of 'port(10)'), within the containing chassis.

            The enumeration 'macAddress(3)' represents a port identifier
            based on a unicast source address (encoded in network
            byte order and IEEE 802.3 canonical bit order), which has
            been detected by the agent and associated with a particular
            port (IEEE Std 802).

            The enumeration 'networkAddress(4)' represents a port
            identifier based on a network address, detected by the agent
            and associated with a particular port.

            The enumeration 'interfaceName(5)' represents a port
            identifier based on the ifName MIB object, defined in IETF
            RFC 2863.

            The enumeration 'agentCircuitId(6)' represents a port
            identifier based on the agent-local identifier of the circuit
            (defined in IETF RFC 3046), detected by the agent and associated
            with a particular port.

            The enumeration 'local(7)' represents a port identifier
            based on a value locally assigned."

    SYNTAX  INTEGER {
            interfaceAlias(1),
            portComponent(2),
            macAddress(3),
            networkAddress(4),
            interfaceName(5),
            agentCircuitId(6),
            local(7)
    }

LldpV2PortId ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "1x:"
    STATUS      current
    DESCRIPTION
            "This TC describes the format of a port identifier string.
            Objects of this type are always used with an associated
```

```
            LldpPortIdSubtype object, which identifies the format of the
            particular LldpPortId object instance.

            If the associated LldpPortIdSubtype object has a value of
            'interfaceAlias(1)', then the octet string identifies a
            particular instance of the ifAlias object (defined in IETF
            RFC 2863). If the particular ifAlias object does not contain
            any values, another port identifier type should be used.

            If the associated LldpPortIdSubtype object has a value of
            'portComponent(2)', then the octet string identifies a
            particular instance of the entPhysicalAlias object (defined
            in IETF RFC 6933) for a port or backplane component.

            If the associated LldpPortIdSubtype object has a value of
            'macAddress(3)', then this string identifies a particular
            unicast source address (encoded in network byte order
            and IEEE 802.3 canonical bit order) associated with the port
            (IEEE Std 802).

            If the associated LldpPortIdSubtype object has a value of
            'networkAddress(4)', then this string identifies a network
            address associated with the port. The first octet contains
            the IANA AddressFamilyNumbers enumeration value for the
            specific address type, and octets 2 through N contain the
            networkAddress address value in network byte order.

            If the associated LldpPortIdSubtype object has a value of
            'interfaceName(5)', then the octet string identifies a
            particular instance of the ifName object (defined in IETF
            RFC 2863). If the particular ifName object does not contain
            any values, another port identifier type should be used.

            If the associated LldpPortIdSubtype object has a value of
            'agentCircuitId(6)', then this string identifies a agent-local
            identifier of the circuit (defined in IETF RFC 3046).

            If the associated LldpPortIdSubtype object has a value of
            'local(7)', then this string identifies a locally
            assigned port ID."
    SYNTAX      OCTET STRING (SIZE (1..255))

LldpV2ManAddrIfSubtype ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
            "This TC defines an enumeration value that identifies
            the interface numbering method used for defining the
            interface number associated with a management address.
            An object with this syntax defines the format of an
            interface number object.

            The enumeration 'unknown(1)' represents the case where the
            interface is not known. In this case, the corresponding
            interface number is of zero length.

            The enumeration 'ifIndex(2)' represents interface identifier
            based on the ifIndex MIB object.

            The enumeration 'systemPortNumber(3)' represents interface
```

```
            identifier based on the system port numbering convention."
    REFERENCE
            "8.5.9.5"

    SYNTAX  INTEGER {
            unknown(1),
            ifIndex(2),
            systemPortNumber(3)
    }

LldpV2ManAddress ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "1x:"
    STATUS      current
    DESCRIPTION
            "The value of a management address associated with the LLDP
            agent that may be used to reach higher layer entities to
            assist discovery by network management.

            It should be noted that appropriate security credentials,
            such as SNMP engineId, may be required to access the LLDP
            agent using a management address. These necessary credentials
            should be known by the network management and the objects
            associated with the credentials are not included in the
            LLDP agent."
    SYNTAX      OCTET STRING (SIZE (1..31))

LldpV2SystemCapabilitiesMap ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
            "This TC describes the system capabilities.

            The bit 'other(0)' indicates that the system has capabilities
            other than those listed below.

            The bit 'repeater(1)' indicates that the system has repeater
            capability.

            The bit 'bridge(2)' indicates that the system has bridge
            capability.

            The bit 'wlanAccessPoint(3)' indicates that the system has
            WLAN access point capability.

            The bit 'router(4)' indicates that the system has router
            capability.

            The bit 'telephone(5)' indicates that the system has telephone
            capability.

            The bit 'docsisCableDevice(6)' indicates that the system has
            DOCSIS Cable Device capability (IETF RFC 4639 & 2670).

            The bit 'stationOnly(7)' indicates that the system has only
            station capability and nothing else.

            The bit 'cVLANComponent(8)' indicates that the system has
            C-VLAN component functionality.

            The bit 'sVLANComponent(8)' indicates that the system  has
```

```
            S-VLAN component functionality.

            The bit 'twoPortMACRelay(10)' indicates that the system has
            Two-port MAC Relay (TPMR) functionality."
    SYNTAX  BITS {
            other(0),
            repeater(1),
            bridge(2),
            wlanAccessPoint(3),
            router(4),
            telephone(5),
            docsisCableDevice(6),
            stationOnly(7),
            cVLANComponent(8),
            sVLANComponent(9),
            twoPortMACRelay(10)

    }


LldpV2DestAddressTableIndex ::= TEXTUAL-CONVENTION
    DISPLAY-HINT "d"
    STATUS    current
    DESCRIPTION
            "An index value, used as the index to the table of destination
            MAC addresses used both as the destination addresses on
            transmitted LLDPDUs and on received LLDPDUs. This index value
            is also used as a secondary index value in tables indexed
            by fields of type  ifIndex, in order to associate
            a destination address with each row of the table."
    SYNTAX  Unsigned32(1..4096)

LldpV2PowerPortClass ::= TEXTUAL-CONVENTION
    STATUS      current
    DESCRIPTION
            "This TC describes the Power over Ethernet (PoE) port class."
    SYNTAX  INTEGER {
            pClassPSE(1),
            pClassPD(2)
    }

END
```

11.5.2 LLDP MIB module - version 2

In the following MIB module, should any discrepancy between the DESCRIPTION text and the corresponding definition in Clause 9 and Clause 10 occur, the definition in Clause 9 and Clause 10 shall take precedence.

```
LLDP-V2-MIB DEFINITIONS ::= BEGIN
IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE,
    Unsigned32,
    Counter32,
    NOTIFICATION-TYPE
        FROM SNMPv2-SMI
    TimeStamp,
    TruthValue,
    MacAddress,
    RowStatus
        FROM SNMPv2-TC
    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB
    MODULE-COMPLIANCE,
    OBJECT-GROUP,
    NOTIFICATION-GROUP
        FROM SNMPv2-CONF
    TimeFilter,
    ZeroBasedCounter32
        FROM RMON2-MIB
    AddressFamilyNumbers
        FROM IANA-ADDRESS-FAMILY-NUMBERS-MIB
    ifGeneralInformationGroup,
    InterfaceIndex
        FROM IF-MIB
    LldpV2ChassisIdSubtype,
    LldpV2ChassisId,
    LldpV2PortIdSubtype,
    LldpV2PortId,
    LldpV2ManAddrIfSubtype,
    LldpV2ManAddress,
    LldpV2SystemCapabilitiesMap,
    LldpV2DestAddressTableIndex,
    ieee802dot1mibs
        FROM LLDP-V2-TC-MIB;

lldpV2MIB MODULE-IDENTITY
    LAST-UPDATED "201603110000Z" -- March 11, 2016
    ORGANIZATION "IEEE 802.1 Working Group"
    CONTACT-INFO
            "WG-URL: http://www.ieee802.org/1/
             WG-EMail: stds-802-1-L@ieee.org

            Contact: IEEE 802.1 Working Group Chair
            Postal:  IEEE Standards Board
                     445 Hoes Lane
                     Piscataway, NJ 08854
                     USA
            E-mail:  stds-802-1-L@ieee.org"
    DESCRIPTION
```

```
            "Management Information Base module for LLDP configuration,
            statistics, local system data and remote systems data
            components.

            This MIB module supports the architecture described in
            Clause 6, where multiple LLDP agents can be associated with
            a single Port, each supporting transmission by means of a
            different MAC address.

            Unless otherwise indicated, the references in this
            MIB module are to IEEE Std 802.1AB-2016.

            Copyright (C) IEEE (2016). This version of this MIB module
            is published as 11.5.2 of IEEE Std 802.1AB-2016;
            see the standard itself for full legal notices."

    REVISION "201603110000Z" -- March 11, 2016

    DESCRIPTION
            "Published as part of the 2016 revision of
            IEEE Std 802.1AB.
            This revision incorporated changes to the MIB to
            address issues identified in maintenance item
            0121 - see http://www.ieee802.org/1/maint.html.
            The changes involved correcting the way that
            lldpV2MessageTxHoldMultiplier is calculated, in
            accordance with 9.2.5.22."


    REVISION "201502160000Z" -- February 16, 2015

    DESCRIPTION
            "Published as part of IEEE Std 802.1AB-2009 Cor-2.
            This corrigendum incorporated changes to the MIB to
            address issues identified in maintenance item
            0121 - see http://www.ieee802.org/1/maint.html."

    REVISION "200906080000Z" -- June 08, 2009

    DESCRIPTION
            "Published as part of IEEE Std 802.1AB-2009 revision.
            This revision incorporated changes to the MIB to
            support the use of LLDP with multiple destination MAC
            addresses."

   ::= { ieee802dot1mibs 13 }

lldpV2Notifications            OBJECT IDENTIFIER ::= { lldpV2MIB 0 }
lldpV2Objects                  OBJECT IDENTIFIER ::= { lldpV2MIB 1 }
lldpV2Conformance              OBJECT IDENTIFIER ::= { lldpV2MIB 2 }

--
-- LLDP MIB Objects
--

lldpV2Configuration            OBJECT IDENTIFIER ::= { lldpV2Objects 1 }
lldpV2Statistics               OBJECT IDENTIFIER ::= { lldpV2Objects 2 }
lldpV2LocalSystemData          OBJECT IDENTIFIER ::= { lldpV2Objects 3 }
lldpV2RemoteSystemsData        OBJECT IDENTIFIER ::= { lldpV2Objects 4 }
```

```
lldpV2Extensions              OBJECT IDENTIFIER ::= { lldpV2Objects 5 }

--
-- ********************************************************
--
--                 L L D P    C O N F I G
--
-- ********************************************************
--

lldpV2MessageTxInterval OBJECT-TYPE
    SYNTAX      Unsigned32(5..32768)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The interval at which LLDP frames are transmitted on
            behalf of this LLDP agent.

            The default value for lldpV2MessageTxInterval object is
            30 seconds.

            The value of this object is used as the initial value of
            the lldpV2PortMessageTxInterval object on row creation in
            the lldpV2PortConfigTableV2.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.7"
    DEFVAL     { 30 }
    ::= { lldpV2Configuration 1 }

lldpV2MessageTxHoldMultiplier OBJECT-TYPE
    SYNTAX      Unsigned32(2..10)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The time to live value expressed as a multiple of the
            lldpV2MessageTxInterval object. The actual time to live
            value used in LLDP frames, transmitted on behalf of this
            LLDP agent, can be expressed by the following formula:
            TTL = min(65535,
            (lldpV2MessageTxInterval*lldpV2MessageTxHoldMultiplier)+1)
            For example, if the value of lldpV2MessageTxInterval is
            '30', and the value of lldpV2MessageTxHoldMultiplier
            is '4', then the value '121' is encoded in the
            TTL field in the LLDP header.

            The default value for lldpV2MessageTxHoldMultiplier
            object is 4.

            The value of this object is used as the initial value of
            the lldpV2PortMessageTxHoldMultiplier object on row creation in
            the lldpV2PortConfigTableV2.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
```

```
            "9.2.5.6"
    DEFVAL      { 4 }
    ::= { lldpV2Configuration 2 }

lldpV2ReinitDelay OBJECT-TYPE
    SYNTAX      Unsigned32(1..10)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The lldpV2ReinitDelay indicates the delay (in units of
            seconds) from when lldpPortConfigAdminStatus object of a
            particular port becomes 'disabled' until re-initialization
            is attempted.

            The default value for lldpV2ReinitDelay is 2 s.

            The value of this object is used as the initial value of
            the lldpV2PortReinitDelay object on row creation in
            the lldpV2PortConfigTableV2.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.10"
    DEFVAL      { 2 }
    ::= { lldpV2Configuration 3 }

lldpV2NotificationInterval OBJECT-TYPE
    SYNTAX      Unsigned32(5..3600)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object controls the interval between transmission of
            LLDP notifications during normal transmission periods.

            The value of this object is used as the initial value of
            the lldpV2PortNotificationInterval object on row creation in
            the lldpV2PortConfigTableV2.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    DEFVAL { 30 }
    ::= { lldpV2Configuration 4 }

lldpV2TxCreditMax OBJECT-TYPE
    SYNTAX Unsigned32(1..100)
    UNITS "PDUs"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
            "The maximum number of consecutive LLDPDUs that can be
            transmitted at any time.

            The default value for lldpV2TxCreditMax object is 5 PDUs.

            The value of this object is used as the initial value of
            the lldpV2PortTxCreditMax object on row creation in
```

```
            the lldpV2PortConfigTableV2.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.17"
    DEFVAL { 5 }
    ::= { lldpV2Configuration 5 }

lldpV2MessageFastTx OBJECT-TYPE
    SYNTAX Unsigned32(1..3600)
    UNITS "seconds"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
                "The interval at which LLDP frames are transmitted on
                behalf of this LLDP agent during fast transmission period
                (e.g., when a new neighbor is detected).

                The default value for lldpV2MessageFastTx object is
                1 second.

                The value of this object is used as the initial value of
                the lldpV2PortMessageFastTx object on row creation in
                the lldpV2PortConfigTableV2.

                The value of this object is restored from non-volatile
                storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.5"
    DEFVAL { 1 }
    ::= { lldpV2Configuration 6 }

lldpV2TxFastInit OBJECT-TYPE
    SYNTAX Unsigned32(1..8)
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
                "The initial value used to initialize the txFast variable
                which determines the number of transmissions that are
                made in fast transmission mode.

                The default value for lldpV2TxFastInit object is 4.

                The value of this object is used as the initial value of
                the lldpV2PortTxFastInit object on row creation in
                the lldpV2PortConfigTableV2.

                The value of this object is restored from non-volatile
                storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.19"
    DEFVAL { 4 }
    ::= { lldpV2Configuration 7 }
--
-- lldpV2PortConfigTable: LLDP configuration indexed on a per port,
-- per destination address basis. The  ifIndex, coupled with an
-- index into the lldpDestAddressTable, is used to index per port
-- per destination MAC address.
```

```
-- ***This table and its associated objects are now deprecated
-- and replaced by lldpV2PortConfigTableV2.***
--

lldpV2PortConfigTable   OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2PortConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      deprecated
    DESCRIPTION
            "The table that controls LLDP frame transmission on individual
            ports that uses particular destination MAC addresses."
    ::= { lldpV2Configuration 8 }

lldpV2PortConfigEntry   OBJECT-TYPE
    SYNTAX      LldpV2PortConfigEntry
    MAX-ACCESS  not-accessible
    STATUS      deprecated
    DESCRIPTION
            "LLDP configuration information for a particular port and
            destination MAC address.

            This configuration parameter controls the transmission and
            the reception of LLDP frames on those interface/address
            combinations whose rows are created in this table.

            Rows in this table can only be created for MAC addresses
            that can validly be used in association with the type of
            interface concerned, as defined by Table 7-2.

            The contents of this table is persistent across
            re-initializations or re-boots."
     INDEX  { lldpV2PortConfigIfIndex,
              lldpV2PortConfigDestAddressIndex  }
    ::= { lldpV2PortConfigTable 1 }

LldpV2PortConfigEntry ::= SEQUENCE {
      lldpV2PortConfigIfIndex             InterfaceIndex,
      lldpV2PortConfigDestAddressIndex    LldpV2DestAddressTableIndex,
      lldpV2PortConfigAdminStatus         INTEGER,
      lldpV2PortConfigNotificationEnable  TruthValue,
      lldpV2PortConfigTLVsTxEnable        BITS }

lldpV2PortConfigIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      deprecated
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index into
            the interfaces MIB.

            The value of this object is used as an index to the
            lldpV2PortConfigTable."
    ::= { lldpV2PortConfigEntry 1 }

lldpV2PortConfigDestAddressIndex   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      deprecated
```

```
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpV2PortConfigTable."
    ::= { lldpV2PortConfigEntry 2 }

lldpV2PortConfigAdminStatus  OBJECT-TYPE
    SYNTAX INTEGER {
       txOnly(1),
       rxOnly(2),
       txAndRx(3),
       disabled(4)
    }
    MAX-ACCESS read-write
    STATUS     deprecated
    DESCRIPTION
            "The administratively desired status of the local LLDP agent.

            If the associated lldpV2PortConfigAdminStatus object is
            set to a value of 'txOnly(1)', then LLDP agent transmits
            LLDPframes on this port and it does not store any
            information about the remote systems connected.

            If the associated lldpV2PortConfigAdminStatus object is
            set to a value of 'rxOnly(2)', then the LLDP agent
            receives, but it does not transmit LLDP frames on this port.

            If the associated lldpV2PortConfigAdminStatus object is set
            to a value of 'txAndRx(3)', then the LLDP agent transmits
            and receives LLDP frames on this port.

            If the associated lldpV2PortConfigAdminStatus object is set
            to a value of 'disabled(4)', then LLDP agent does not
            transmit or receive LLDP frames on this port. If there is
            remote systems information that is received on this port
            and stored in other tables, before the port's
            lldpV2PortConfigAdminStatus becomes disabled, then that
            information is deleted."
    REFERENCE
            "9.2.5.1"
    DEFVAL  { txAndRx }
   ::= { lldpV2PortConfigEntry 3 }

lldpV2PortConfigNotificationEnable OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-write
    STATUS     deprecated
    DESCRIPTION
            "The lldpV2PortConfigNotificationEnable controls, on a per
            agent basis, whether or not notifications from the agent
            are enabled. The value true(1) means that notifications are
            enabled; the value false(2) means that they are not."
    DEFVAL  { false }
   ::= { lldpV2PortConfigEntry 4 }
```

```
lldpV2PortConfigTLVsTxEnable OBJECT-TYPE
    SYNTAX      BITS {
            portDesc(0),
            sysName(1),
            sysDesc(2),
            sysCap(3)
    }
    MAX-ACCESS  read-write
    STATUS      deprecated
    DESCRIPTION
            "The lldpV2PortConfigTLVsTxEnable, defined as a bitmap,
            includes the basic set of LLDP TLVs whose transmission is
            allowed on the local LLDP agent by the network management.
            Each bit in the bitmap corresponds to a TLV type associated
            with a specific optional TLV.

            It should be noted that the organizationally-specific TLVs
            are excluded from the lldpV2PortConfigTLVsTxEnable bitmap.

            LLDP Organization Specific Information Extension MIBs should
            have similar configuration objects to control transmission
            of their organizationally defined TLVs.

            The bit 'portDesc(0)' indicates that LLDP agent should
            transmit 'Port Description TLV'.

            The bit 'sysName(1)' indicates that LLDP agent should transmit
            'System Name TLV'.

            The bit 'sysDesc(2)' indicates that LLDP agent should transmit
            'System Description TLV'.

            The bit 'sysCap(3)' indicates that LLDP agent should transmit
            'System Capabilities TLV'.

            There is no bit reserved for the management address TLV type
            since transmission of management address TLVs are controlled
            by another object, lldpV2ConfigManAddrTable.

            The default value for lldpV2PortConfigTLVsTxEnable object is
            empty set, which means no enumerated values are set.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.1.2.1"
    DEFVAL  { { } }
    ::= { lldpV2PortConfigEntry 5 }

--
-- lldpV2PortConfigTableV2: LLDP configuration indexed on a per port,
-- per destination address basis. The  ifIndex, coupled with an
-- index into the lldpDestAddressTable, is used to index per port
-- per destination MAC address.
--
-- V2 extends the original table definition to include per-port
-- per-MAC address parameters msgTxInterval, msgTxHold, reinitDelay,
-- notificationInterval, txCreditMax, msgFastTx, and txFastInit.
--
```

```
lldpV2PortConfigTableV2   OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2PortConfigEntryV2
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The table that controls LLDP frame transmission on individual
            ports and using particular destination MAC addresses."
    ::= { lldpV2Configuration 11 }

lldpV2PortConfigEntryV2   OBJECT-TYPE
    SYNTAX      LldpV2PortConfigEntryV2
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "LLDP configuration information for a particular port and
            destination MAC address.

            This configuration parameter controls the transmission and
            the reception of LLDP frames on those interface/address
            combinations whose rows are created in this table.

            Rows in this table can only be created for MAC addresses
            that can validly be used in association with the type of
            interface concerned, as defined by Table 7-2.

            The contents of this table is persistent across
            re-initializations or re-boots."
     INDEX  { lldpV2PortConfigIfIndexV2,
              lldpV2PortConfigDestAddressIndexV2  }
    ::= { lldpV2PortConfigTableV2 1 }

LldpV2PortConfigEntryV2 ::= SEQUENCE {
      lldpV2PortConfigIfIndexV2             InterfaceIndex,
      lldpV2PortConfigDestAddressIndexV2    LldpV2DestAddressTableIndex,
      lldpV2PortConfigAdminStatusV2         INTEGER,
      lldpV2PortMessageTxInterval           Unsigned32,
      lldpV2PortMessageTxHoldMultiplier     Unsigned32,
      lldpV2PortReinitDelay                 Unsigned32,
      lldpV2PortNotificationInterval        Unsigned32,
      lldpV2PortTxCreditMax                 Unsigned32,
      lldpV2PortMessageFastTx               Unsigned32,
      lldpV2PortTxFastInit                  Unsigned32,
      lldpV2PortConfigNotificationEnableV2  TruthValue,
      lldpV2PortConfigTLVsTxEnableV2        BITS }

lldpV2PortConfigIfIndexV2   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index into
            the interfaces MIB.

            The value of this object is used as an index to the
            lldpV2PortConfigTable."
    ::= { lldpV2PortConfigEntryV2 1 }
```

```
lldpV2PortConfigDestAddressIndexV2   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpV2PortConfigTable."
    ::= { lldpV2PortConfigEntryV2 2 }

lldpV2PortConfigAdminStatusV2  OBJECT-TYPE
    SYNTAX INTEGER {
       txOnly(1),
       rxOnly(2),
       txAndRx(3),
       disabled(4)
    }
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
            "The administratively desired status of the local LLDP agent.

            If the associated lldpV2PortConfigAdminStatus object is
            set to a value of 'txOnly(1)', then LLDP agent transmits
            LLDPframes on this port and it does not store any
            information about the remote systems connected.

            If the associated lldpV2PortConfigAdminStatus object is
            set to a value of 'rxOnly(2)', then the LLDP agent
            receives, but it does not transmit, LLDP frames on this port.

            If the associated lldpV2PortConfigAdminStatus object is set
            to a value of 'txAndRx(3)', then the LLDP agent transmits
            and receives LLDP frames on this port.

            If the associated lldpV2PortConfigAdminStatus object is set
            to a value of 'disabled(4)', then LLDP agent does not
            transmit or receive LLDP frames on this port. If there is
            remote systems information that is received on this port
            and stored in other tables, before the port's
            lldpV2PortConfigAdminStatus becomes disabled, then that
            information is deleted."
    REFERENCE
            "9.2.5.1"
    DEFVAL  { txAndRx }
   ::= { lldpV2PortConfigEntryV2 3 }

lldpV2PortMessageTxInterval OBJECT-TYPE
    SYNTAX      Unsigned32(5..32768)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The interval at which LLDP frames are transmitted on
            behalf of this LLDP agent.
```

```
            This object takes its initial value from the
            lldpV2MessageTxInterval object on table row creation.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.7"
    DEFVAL     { 30 }
    ::= { lldpV2PortConfigEntryV2 4 }

lldpV2PortMessageTxHoldMultiplier OBJECT-TYPE
    SYNTAX      Unsigned32(2..10)
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The time to live value expressed as a multiple of the
            lldpV2MessageTxInterval object. The actual time to live
            value used in LLDP frames, transmitted on behalf of this
            LLDP agent, can be expressed by the following formula:
            TTL = min(65535,
            (lldpV2MessageTxInterval*lldpV2MessageTxHoldMultiplier)+1)
            For example, if the value of lldpV2MessageTxInterval is
            '30', and the value of lldpV2MessageTxHoldMultiplier
            is '4', then the value '121' is encoded in the
            TTL field in the LLDP header.

            This object takes its initial value from the
            lldpV2PortMessageTxHoldMultiplier object on table row creation.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.6"
    DEFVAL      { 4 }
    ::= { lldpV2PortConfigEntryV2 5 }

lldpV2PortReinitDelay OBJECT-TYPE
    SYNTAX      Unsigned32(1..10)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The lldpV2ReinitDelay indicates the delay (in units of
            seconds) from when lldpPortConfigAdminStatus object of a
            particular port becomes 'disabled' until re-initialization
            is attempted.

            This object takes its initial value from the
            lldpV2PortReinitDelay object on table row creation.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.10"
    DEFVAL      { 2 }
    ::= { lldpV2PortConfigEntryV2 6 }

lldpV2PortNotificationInterval OBJECT-TYPE
```

```
    SYNTAX      Unsigned32(5..3600)
    UNITS       "seconds"
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "This object controls the interval between transmission of
            LLDP notifications during normal transmission periods.

            This object takes its initial value from the
            lldpV2PortNotificationInterval object on table row creation.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    DEFVAL { 30 }
    ::= { lldpV2PortConfigEntryV2 7 }

lldpV2PortTxCreditMax OBJECT-TYPE
    SYNTAX Unsigned32(1..100)
    UNITS "PDUs"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
            "The maximum number of consecutive LLDPDUs that can be
            transmitted at any time.

            This object takes its initial value from the
            lldpV2PortTxCreditMax object on table row creation.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.17"
    DEFVAL { 5 }
    ::= { lldpV2PortConfigEntryV2 8 }

lldpV2PortMessageFastTx OBJECT-TYPE
    SYNTAX Unsigned32(1..3600)
    UNITS "seconds"
    MAX-ACCESS read-write
    STATUS current
    DESCRIPTION
                "The interval at which LLDP frames are transmitted on
                behalf of this LLDP agent during fast transmission period
                (e.g., when a new neighbor is detected).

                This object takes its initial value from the
                lldpV2PortMessageFastTx object on table row creation.

                The value of this object is restored from non-volatile
                storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.5"
    DEFVAL { 1 }
    ::= { lldpV2PortConfigEntryV2 9 }

lldpV2PortTxFastInit OBJECT-TYPE
    SYNTAX Unsigned32(1..8)
    MAX-ACCESS read-write
    STATUS current
```

```
    DESCRIPTION
                "The initial value used to initialize the txFast variable
                which determines the number of transmissions that are
                made in fast transmission mode.

                This object takes its initial value from the
                lldpV2PortTxFastInit object on table row creation.

                The value of this object is restored from non-volatile
                storage after a re-initialization of the management system."
    REFERENCE
            "9.2.5.19"
    DEFVAL { 4 }
    ::= { lldpV2PortConfigEntryV2 10 }


lldpV2PortConfigNotificationEnableV2 OBJECT-TYPE
    SYNTAX     TruthValue
    MAX-ACCESS read-write
    STATUS     current
    DESCRIPTION
            "The lldpV2PortConfigNotificationEnableV2 controls, on a per
            agent basis, whether or not notifications from the agent
            are enabled. The value true(1) means that notifications are
            enabled; the value false(2) means that they are not."
    DEFVAL  { false }
   ::= { lldpV2PortConfigEntryV2 11 }

lldpV2PortConfigTLVsTxEnableV2 OBJECT-TYPE
    SYNTAX      BITS {
            portDesc(0),
            sysName(1),
            sysDesc(2),
            sysCap(3)
    }
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION
            "The lldpV2PortConfigTLVsTxEnableV2, defined as a bitmap,
            includes the basic set of LLDP TLVs whose transmission is
            allowed on the local LLDP agent by the network management.
            Each bit in the bitmap corresponds to a TLV type associated
            with a specific optional TLV.

            It should be noted that the organizationally-specific TLVs
            are excluded from the lldpV2PortConfigTLVsTxEnable bitmap.

            LLDP Organization Specific Information Extension MIBs should
            have similar configuration objects to control transmission
            of their organizationally defined TLVs.

            The bit 'portDesc(0)' indicates that LLDP agent should
            transmit 'Port Description TLV'.

            The bit 'sysName(1)' indicates that LLDP agent should transmit
            'System Name TLV'.

            The bit 'sysDesc(2)' indicates that LLDP agent should transmit
            'System Description TLV'.
```

```
            The bit 'sysCap(3)' indicates that LLDP agent should transmit
            'System Capabilities TLV'.

            There is no bit reserved for the management address TLV type
            since transmission of management address TLVs are controlled
            by another object, lldpV2ConfigManAddrTable.

            The default value for lldpV2PortConfigTLVsTxEnable object is
            empty set, which means no enumerated values are set.

            The value of this object is restored from non-volatile
            storage after a re-initialization of the management system."
    REFERENCE
            "9.1.2.1"
    DEFVAL  { { } }
    ::= { lldpV2PortConfigEntryV2 12 }


--
-- lldpV2DestAddressTable: Destination MAC addresses used by LLDP
--

lldpV2DestAddressTable   OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2DestAddressTableEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The table that contains the set of MAC addresses used
            by LLDP for transmission and reception of LLDPDUs."
    ::= { lldpV2Configuration 9 }

lldpV2DestAddressTableEntry   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Destination MAC address information for LLDP.

            This configuration parameter identifies a MAC address
            corresponding to a LldpV2DestAddressTableIndex value.

            Rows in this table are created as necessary, to support
            MAC addresses needed by other tables in the MIB that
            are indexed by MAC address.

            A given row in this table cannot be deleted if the MAC
            address table index value is in use in any other table
            in the MIB.

            The contents of this table are persistent across
            re-initializations or re-boots."
     INDEX  { lldpV2AddressTableIndex  }
    ::= { lldpV2DestAddressTable 1 }

LldpV2DestAddressTableEntry ::= SEQUENCE {
      lldpV2AddressTableIndex      LldpV2DestAddressTableIndex,
      lldpV2DestMacAddress         MacAddress  }
```

```
lldpV2AddressTableIndex   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry.

            The value of this object is used as an index to the
            lldpV2DestAddressTable."
    ::= { lldpV2DestAddressTableEntry 1 }

lldpV2DestMacAddress   OBJECT-TYPE
    SYNTAX      MacAddress
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The MAC address associated with this entry.

            The octet string identifies an individual or a group
            MAC address that is in use by LLDP as a destination
            MAC address.

            The MAC address is encoded in the octet string in
            canonical format (see IEEE Std 802)."
    ::= { lldpV2DestAddressTableEntry 2 }

--
-- lldpV2ManAddrConfigTxPortsTable : selection of management addresses
-- to be transmitted on a specified set of port/destination
-- address pairs.
--

lldpV2ManAddrConfigTxPortsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2ManAddrConfigTxPortsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The table that controls selection of LLDP management address
            TLV instances to be transmitted on individual port/
            destination address pairs."
    ::= { lldpV2Configuration 10 }

lldpV2ManAddrConfigTxPortsEntry  OBJECT-TYPE
    SYNTAX      LldpV2ManAddrConfigTxPortsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "LLDP configuration information that specifies the set
            of port/destination address pairs on which the local
            system management address instance is transmitted.

            Each active lldpManAddrConfigTxPortsTableV2Entry is
            restored from non-volatile storage and re-created
            after a re-initialization of the management system."
    INDEX {
            lldpV2ManAddrConfigIfIndex,
            lldpV2ManAddrConfigDestAddressIndex,
            lldpV2ManAddrConfigLocManAddrSubtype,
```

```
            lldpV2ManAddrConfigLocManAddr }
    ::= { lldpV2ManAddrConfigTxPortsTable 1 }

LldpV2ManAddrConfigTxPortsEntry  ::= SEQUENCE {
            lldpV2ManAddrConfigIfIndex            InterfaceIndex,
            lldpV2ManAddrConfigDestAddressIndex   LldpV2DestAddressTableIndex,
            lldpV2ManAddrConfigLocManAddrSubtype  AddressFamilyNumbers,
            lldpV2ManAddrConfigLocManAddr         LldpV2ManAddress,
            lldpV2ManAddrConfigTxEnable           TruthValue,
            lldpV2ManAddrConfigRowStatus          RowStatus
}

lldpV2ManAddrConfigIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index into
            the interfaces MIB.

            The value of this object is used as an index to the
            lldpV2PortConfigTable.

            The value in this column of the table MUST match
            the IfIndex value specified in the BridgePort table."
    ::= { lldpV2ManAddrConfigTxPortsEntry 1 }

lldpV2ManAddrConfigDestAddressIndex   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpV2PortConfigTable."
    ::= { lldpV2ManAddrConfigTxPortsEntry 2 }


lldpV2ManAddrConfigLocManAddrSubtype  OBJECT-TYPE
    SYNTAX      AddressFamilyNumbers
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The type of management address identifier encoding used in
            the associated 'lldpLocManagmentAddr' object.

            It should be noted that only a subset of the possible
            address encodings enumerated in AddressFamilyNumbers
            are appropriate for use as a LLDP management
            address, either because some are just not applicable or
            because the maximum size of a LldpV2ManAddress octet string
            would prevent the use of some address identifier encodings."
    REFERENCE
            "8.5.9.3"
```

```
    ::= { lldpV2ManAddrConfigTxPortsEntry 3 }

lldpV2ManAddrConfigLocManAddr  OBJECT-TYPE
    SYNTAX      LldpV2ManAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The string value used to identify the management address
            component associated with the local system. The purpose of
            this address is to contact the management entity."
    REFERENCE
            "8.5.9.4"
    ::= { lldpV2ManAddrConfigTxPortsEntry 4 }



lldpV2ManAddrConfigTxEnable  OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-create
    STATUS        current
    DESCRIPTION
            "A Boolean controlling the transmission of system
             management address instance for the specified port,
             destination, subtype, and MAN address used to index
             this table. If set to the default value of false,
             no transmission occurs. If set to true, the
             appropriate information is transmitted out of the
             port specified in the row's index."
    REFERENCE
            "9.1.2.1"
    DEFVAL  { false }     -- not transmitted
    ::= { lldpV2ManAddrConfigTxPortsEntry 5 }

lldpV2ManAddrConfigRowStatus OBJECT-TYPE
    SYNTAX RowStatus
    MAX-ACCESS read-create
    STATUS current
    DESCRIPTION
        "Indicates the status of an entry in this table, and is used
        to create/delete entries.
        The corresponding instances of the following objects
        must be set before this object can be made active(1):
           lldpV2ManAddrConfigDestAddressIndex
           lldpV2ManAddrConfigLocManAddrSubtype
           lldpV2ManAddrConfigLocManAddr
           lldpV2ManAddrConfigTxEnable

        The corresponding instances of the following objects
        may not be changed while this object is active(1):
           lldpV2ManAddrConfigDestAddressIndex
           lldpV2ManAddrConfigLocManAddrSubtype
           lldpV2ManAddrConfigLocManAddr "
::= { lldpV2ManAddrConfigTxPortsEntry 6 }

--
--  ********************************************************
--
--                  L L D P    S T A T S
--
```

```
--  ********************************************************
--
-- LLDP Stats Group

lldpV2StatsRemTablesLastChangeTime OBJECT-TYPE
    SYNTAX      TimeStamp
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The value of sysUpTime object (defined in IETF RFC 3418)
            at the time an entry is created, modified, or deleted in the
            in tables associated with the lldpV2RemoteSystemsData objects
            and all LLDP extension objects associated with remote systems.

            An NMS can use this object to reduce polling of the
            lldpV2RemoteSystemsData objects."
    ::= { lldpV2Statistics 1 }

lldpV2StatsRemTablesInserts OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "table entries"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number of times the complete set of information
            advertised by a particular MSAP has been inserted into tables
            contained in lldpV2RemoteSystemsData and lldpV2Extensions objects.

            The complete set of information received from a particular
            MSAP should be inserted into related tables. If partial
            information cannot be inserted for a reason such as lack
            of resources, all of the complete set of information should
            be removed.

            This counter should be incremented only once after the
            complete set of information is successfully recorded
            in all related tables. Any failures during inserting
            information set that result in deletion of previously
            inserted information should not trigger any changes in
            lldpV2StatsRemTablesInserts since the insert is not completed
            yet or in lldpStatsRemTablesDeletes since the deletion
            would only be a partial deletion. If the failure was the
            result of lack of resources, the lldpStatsRemTablesDrops
            counter should be incremented once."
    ::= { lldpV2Statistics 2 }

lldpV2StatsRemTablesDeletes   OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "table entries"
    MAX-ACCESS  read-only
    STATUS      current

    DESCRIPTION
            "The number of times the complete set of information
            advertised by a particular MSAP has been deleted from
            tables contained in lldpV2RemoteSystemsData and lldpV2Extensions
            objects.

            This counter should be incremented only once when the
```

```
            complete set of information is completely deleted from all
            related tables. Partial deletions, such as deletion of
            rows associated with a particular MSAP from some tables,
            but not from all tables are not allowed, thus should not
            change the value of this counter."
    ::= { lldpV2Statistics 3 }

lldpV2StatsRemTablesDrops  OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "table entries"
    MAX-ACCESS  read-only

    STATUS      current
    DESCRIPTION
            "The number of times the complete set of information
            advertised by a particular MSAP could not be entered into
            tables contained in lldpV2RemoteSystemsData and lldpV2Extensions
            objects because of insufficient resources."
    ::= { lldpV2Statistics 4 }

lldpV2StatsRemTablesAgeouts   OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
    UNITS       "table entries"
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The number of times the complete set of information
            advertised by a particular MSAP has been deleted from tables
            contained in lldpV2RemoteSystemsData and lldpV2Extensions objects
            because the information timeliness interval has expired.

            This counter should be incremented only once when the complete
            set of information is completely invalidated (aged out)
            from all related tables. Partial ageing, similar to deletion
            case, is not allowed, and thus, should not change the value
            of this counter."
    ::= { lldpV2Statistics 5 }

--
-- TX statistics
-- Indexed by port (via ifIndex) and
-- destination MAC address.
--

lldpV2StatsTxPortTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2StatsTxPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A table containing LLDP transmission statistics for
            individual port/destination address combinations.
            Entries are not required to exist in
            this table while the lldpPortConfigEntry object is equal to
            'disabled(4)'."
    ::= { lldpV2Statistics 6 }

lldpV2StatsTxPortEntry   OBJECT-TYPE
     SYNTAX      LldpV2StatsTxPortEntry
     MAX-ACCESS  not-accessible
```

```
    STATUS      current
    DESCRIPTION
            "LLDP frame transmission statistics for a particular port
            and destination MAC address.
            The port is contained in the same chassis as the
            LLDP agent.

            All counter values in a particular entry shall be
            maintained on a continuing basis and shall not be deleted
            upon expiration of rxInfoTTL timing counters in the LLDP
            remote systems MIB of the receipt of a shutdown frame from
            a remote LLDP agent.

            All statistical counters associated with a particular
            port on the local LLDP agent become frozen whenever the
            adminStatus is disabled for the same port.

            Rows in this table can only be created for MAC addresses
            that can validly be used in association with the type of
            interface concerned, as defined by Table 7-2."
    INDEX  { lldpV2StatsTxIfIndex,
             lldpV2StatsTxDestMACAddress }
    ::= { lldpV2StatsTxPortTable 1 }

LldpV2StatsTxPortEntry ::= SEQUENCE {
      lldpV2StatsTxIfIndex               InterfaceIndex,
      lldpV2StatsTxDestMACAddress        LldpV2DestAddressTableIndex,
      lldpV2StatsTxPortFramesTotal       Counter32,
      lldpV2StatsTxLLDPDULengthErrors    Counter32  }

lldpV2StatsTxIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index
            into the interfaces MIB.

            The value of this object is used as an index to the
            lldpV2StatsTxPortTable."
    ::= { lldpV2StatsTxPortEntry 1 }

lldpV2StatsTxDestMACAddress   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpV2StatsTxPortTable."
    ::= { lldpV2StatsTxPortEntry 2 }

lldpV2StatsTxPortFramesTotal  OBJECT-TYPE
    SYNTAX        Counter32
```

```
    UNITS         "LLDP frames"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
            "The number of LLDP frames transmitted by this LLDP agent
            on the indicated port to the destination MAC address
            associated with this row of the table."
    REFERENCE
            "9.2.6.5"
    ::= { lldpV2StatsTxPortEntry 3 }

lldpV2StatsTxLLDPDULengthErrors  OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "LLDP frames"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
            "The number of LLDPDU Length Errors recorded for the Port."
    REFERENCE
            "9.2.6.8"
    ::= { lldpV2StatsTxPortEntry 4 }

--
-- lldpV2StatsRxPortTable - RX statistics
-- This table is indexed by ifIndex and destination MAC address.
--

lldpV2StatsRxPortTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2StatsRxPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A table containing LLDP reception statistics for individual
            ports and destination MAC addresses.
            Entries are not required to exist in this table while
            the lldpPortConfigEntry object is equal to 'disabled(4)'."
    ::= { lldpV2Statistics 7 }

lldpV2StatsRxPortEntry   OBJECT-TYPE
     SYNTAX      LldpV2StatsRxPortEntry
     MAX-ACCESS  not-accessible
     STATUS      current
     DESCRIPTION
             "LLDP frame reception statistics for a particular port.
             The port is contained in the same chassis as the
             LLDP agent.

             All counter values in a particular entry shall be
             maintained on a continuing basis and shall not be deleted
             upon expiration of rxInfoTTL timing counters in the LLDP
             remote systems MIB of the receipt of a shutdown frame from
             a remote LLDP agent.

             All statistical counters associated with a particular
             port on the local LLDP agent become frozen whenever the
             adminStatus is disabled for the same port.

             Rows in this table can only be created for MAC addresses
             that can validly be used in association with the type of
```

```
             interface concerned, as defined by Table 7-2.

             The contents of this table is persistent across
             re-initializations or re-boots."
     INDEX  { lldpV2StatsRxDestIfIndex,
              lldpV2StatsRxDestMACAddress }
     ::= { lldpV2StatsRxPortTable 1 }

LldpV2StatsRxPortEntry ::= SEQUENCE {
      lldpV2StatsRxDestIfIndex               InterfaceIndex,
      lldpV2StatsRxDestMACAddress            LldpV2DestAddressTableIndex,
      lldpV2StatsRxPortFramesDiscardedTotal  Counter32,
      lldpV2StatsRxPortFramesErrors          Counter32,
      lldpV2StatsRxPortFramesTotal           Counter32,
      lldpV2StatsRxPortTLVsDiscardedTotal    Counter32,
      lldpV2StatsRxPortTLVsUnrecognizedTotal Counter32,
      lldpV2StatsRxPortAgeoutsTotal          ZeroBasedCounter32
}

lldpV2StatsRxDestIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index
            into the interfaces MIB.

            The value of this object is used as an index to the
            lldpStatsRxPortV2Table."
    ::= { lldpV2StatsRxPortEntry 1 }

lldpV2StatsRxDestMACAddress   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpStatsRxPortV2Table."
    ::= { lldpV2StatsRxPortEntry 2 }


lldpV2StatsRxPortFramesDiscardedTotal OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "LLDP frames"
    MAX-ACCESS read-only
    STATUS     current
    DESCRIPTION
            "The number of LLDP frames received by this LLDP agent on
            the indicated port, and then discarded for any reason.
            This counter can provide an indication that LLDP header
            formatting problems may exist with the local LLDP agent in
            the sending system or that LLDPDU validation problems may
            exist with the local LLDP agent in the receiving system."
```

```
   REFERENCE
           "9.2.6.2"
   ::= { lldpV2StatsRxPortEntry 3 }

lldpV2StatsRxPortFramesErrors  OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "LLDP frames"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
           "The number of invalid LLDP frames received by this LLDP
           agent on the indicated port, while this LLDP agent is enabled."
    REFERENCE
           "9.2.6.3"
    ::= { lldpV2StatsRxPortEntry 4 }

lldpV2StatsRxPortFramesTotal OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "LLDP frames"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
           "The number of valid LLDP frames received by this LLDP agent
           on the indicated port, while this LLDP agent is enabled."
    REFERENCE
           "9.2.6.4"
    ::= { lldpV2StatsRxPortEntry 5 }

lldpV2StatsRxPortTLVsDiscardedTotal OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "TLVs"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
           "The number of LLDP TLVs discarded for any reason by this LLDP
           agent on the indicated port."
    REFERENCE
           "9.2.6.6"
    ::= { lldpV2StatsRxPortEntry 6 }

lldpV2StatsRxPortTLVsUnrecognizedTotal  OBJECT-TYPE
    SYNTAX        Counter32
    UNITS         "TLVs"
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
           "The number of LLDP TLVs received on the given port that
           are not recognized by this LLDP agent on the indicated port.

           An unrecognized TLV is referred to as the TLV whose type value
           is in the range of reserved TLV types (000 1001 - 111 1110)
           in Table 8-1 of IEEE Std 802.1AB-2015. An unrecognized
           TLV may be a basic management TLV from a later LLDP version."
    REFERENCE
           "9.2.6.7"
    ::= { lldpV2StatsRxPortEntry 7 }

lldpV2StatsRxPortAgeoutsTotal   OBJECT-TYPE
    SYNTAX      ZeroBasedCounter32
```

```
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The counter that represents the number of age-outs that
            occurred on a given port. An age-out is the number of
            times the complete set of information advertised by a
            particular MSAP has been deleted from tables contained in
            lldpV2RemoteSystemsData and lldpV2Extensions objects because
            the information timeliness interval has expired.

            This counter is similar to lldpV2StatsRemTablesAgeouts, except
            that the counter is on a per port basis. This enables NMS to
            poll tables associated with the lldpV2RemoteSystemsData objects
            and all LLDP extension objects associated with remote systems
            on the indicated port only.

            This counter is set to zero during agent initialization
            and its value should not be saved in non-volatile storage.

            This counter is incremented only once when the
            complete set of information is invalidated (aged out) from
            all related tables on a particular port. Partial ageing
            is not allowed."
    REFERENCE
            "9.2.6.1"
    ::= { lldpV2StatsRxPortEntry 8 }


--  ********************************************************
--
--          L O C A L    S Y S T E M    D A T A
--
--  ********************************************************

lldpV2LocChassisIdSubtype  OBJECT-TYPE
    SYNTAX      LldpV2ChassisIdSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of encoding used to identify the chassis
            associated with the local system."
    REFERENCE
            "8.5.2.2"
    ::= { lldpV2LocalSystemData 1 }

lldpV2LocChassisId  OBJECT-TYPE
    SYNTAX      LldpV2ChassisId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the chassis component
            associated with the local system."
    REFERENCE
            "8.5.2.3"
    ::= { lldpV2LocalSystemData 2 }

lldpV2LocSysName  OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
            "The string value used to identify the system name of the
            local system. If the local agent supports IETF RFC 3418,
            lldpLocSysName object should have the same value as sysName
            object."
    REFERENCE
            "8.5.6.2"
    ::= { lldpV2LocalSystemData 3 }

lldpV2LocSysDesc  OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the system description
            of the local system. If the local agent supports IETF RFC 3418,
            lldpLocSysDesc object should have the same value as sysDesc
            object."
    REFERENCE
            "8.5.7.2"
    ::= { lldpV2LocalSystemData 4 }

lldpV2LocSysCapSupported OBJECT-TYPE
    SYNTAX      LldpV2SystemCapabilitiesMap
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The bitmap value used to identify which system capabilities
            are supported on the local system."
    REFERENCE
            "8.5.8.1"
    ::= { lldpV2LocalSystemData 5 }

lldpV2LocSysCapEnabled  OBJECT-TYPE
    SYNTAX      LldpV2SystemCapabilitiesMap
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The bitmap value used to identify which system capabilities
            are enabled on the local system."
    REFERENCE
            "8.5.8.2"
    ::= { lldpV2LocalSystemData 6 }


--
-- lldpV2LocPortTable : Port specific Local system data
-- Indexed by  ifIndex.
--

lldpV2LocPortTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2LocPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of information
            associated with the local system known to this agent."
    ::= { lldpV2LocalSystemData 7 }
```

```
lldpV2LocPortEntry OBJECT-TYPE
    SYNTAX      LldpV2LocPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Information about a particular port component.

            Entries may be created and deleted in this table by the
            agent.

            Rows in this table can only be created for MAC addresses
            that can validly be used in association with the type of
            interface concerned, as defined by Table 7-2.

            The contents of this table is persistent across
            re-initializations or re-boots."
    INDEX   { lldpV2LocPortIfIndex }
    ::= { lldpV2LocPortTable 1 }

LldpV2LocPortEntry ::= SEQUENCE {
      lldpV2LocPortIfIndex            InterfaceIndex,
      lldpV2LocPortIdSubtype          LldpV2PortIdSubtype,
      lldpV2LocPortId                 LldpV2PortId,
      lldpV2LocPortDesc               SnmpAdminString
}

lldpV2LocPortIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index
            into the interfaces MIB.

            The value of this object is used as an index to the
            lldpV2LocPortTable."
    ::= { lldpV2LocPortEntry 1 }

lldpV2LocPortIdSubtype  OBJECT-TYPE
    SYNTAX      LldpV2PortIdSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of port identifier encoding used in the associated
            'lldpLocPortId' object."
    REFERENCE
            "8.5.3.2"
    ::= { lldpV2LocPortEntry 2 }

lldpV2LocPortId   OBJECT-TYPE
    SYNTAX      LldpV2PortId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the port component
            associated with a given port in the local system."
    REFERENCE
```

```
            "8.5.3.3"
    ::= { lldpV2LocPortEntry 3 }

lldpV2LocPortDesc   OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the IEEE 802 LAN station's port
            description associated with the local system. If the local
            agent supports IETF RFC 2863, lldpLocPortDesc object should
            have the same value of ifDescr object."
    REFERENCE
            "8.5.5.2"
    ::= { lldpV2LocPortEntry 4 }

--
-- lldpV2LocManAddrTable : Management addresses of the local system
--

lldpV2LocManAddrTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2LocManAddrEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains management address information on the
            local system known to this agent."
    ::= { lldpV2LocalSystemData 8 }

lldpV2LocManAddrEntry OBJECT-TYPE
    SYNTAX      LldpV2LocManAddrEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Management address information about a particular chassis
            component. There may be multiple management addresses
            configured on the system identified by a particular
            lldpLocChassisId. Each management address should have
            distinct 'management address type' (lldpV2LocManAddrSubtype) and
            'management address' (lldpLocManAddr.)

            Entries may be created and deleted in this table by the
            agent.

            Since a variable length octetstring is used as an index
            in a table, the address length is encoded as part of the OID
            (as per IETF RFC 2578)."

    INDEX   { lldpV2LocManAddrSubtype,
              lldpV2LocManAddr }
    ::= { lldpV2LocManAddrTable 1 }

LldpV2LocManAddrEntry ::= SEQUENCE {
      lldpV2LocManAddrSubtype    AddressFamilyNumbers,
      lldpV2LocManAddr           LldpV2ManAddress,
      lldpV2LocManAddrLen        Unsigned32,
      lldpV2LocManAddrIfSubtype  LldpV2ManAddrIfSubtype,
      lldpV2LocManAddrIfId       Unsigned32,
      lldpV2LocManAddrOID        OBJECT IDENTIFIER
```

```
}

lldpV2LocManAddrSubtype  OBJECT-TYPE
    SYNTAX      AddressFamilyNumbers
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The type of management address identifier encoding used in
            the associated 'lldpLocManagmentAddr' object.

            It should be noted that only a subset of the possible
            address encodings enumerated in AddressFamilyNumbers
            are appropriate for use as a LLDP management
            address, either because some are just not applicable or
            because the maximum size of a LldpV2ManAddress octet string
            would prevent the use of some address identifier encodings."
    REFERENCE
            "8.5.9.3"
    ::= { lldpV2LocManAddrEntry 1 }

lldpV2LocManAddr  OBJECT-TYPE
    SYNTAX      LldpV2ManAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The string value used to identify the management address
            component associated with the local system. The purpose of
            this address is to contact the management entity."
    REFERENCE
            "8.5.9.4"
    ::= { lldpV2LocManAddrEntry 2 }

lldpV2LocManAddrLen  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The total length of the management address subtype and the
            management address fields in LLDPDUs transmitted by the
            local LLDP agent.

            The management address length field is needed so that the
            receiving systems that do not implement SNMP are not
            required to implement an Internet Assigned Numbers
            Authority (IANA) family numbers/address length equivalency
            table in order to decode the management address."
    REFERENCE
            "8.5.9.2"
    ::= { lldpV2LocManAddrEntry 3 }

lldpV2LocManAddrIfSubtype  OBJECT-TYPE
    SYNTAX      LldpV2ManAddrIfSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The enumeration value that identifies the interface numbering
            method used for defining the interface number
            (lldpV2LocManAddrIfId), associated with the local system."
    REFERENCE
```

```
            "8.5.9.5"
    ::= { lldpV2LocManAddrEntry 4 }

lldpV2LocManAddrIfId  OBJECT-TYPE
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The integer value used to identify the interface number
            regarding the management address component associated with
            the local system."
    REFERENCE
            "8.5.9.6"
    ::= { lldpV2LocManAddrEntry 5 }

lldpV2LocManAddrOID  OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The OID value used to identify the type of hardware component
            or protocol entity associated with the management address
            advertised by the local system agent."
    REFERENCE
            "8.5.9.8"
    ::= { lldpV2LocManAddrEntry 6 }


--  ********************************************************
--
--          R E M O T E    S Y S T E M S    D A T A
--
--  ********************************************************


--
-- lldpV2RemTable
-- Indexed by ifIndex and destination MAC address.
--

lldpV2RemTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2RemEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one or more rows per physical network
            connection known to this agent. The agent may wish to ensure
            that only one lldpRemEntry is present for each local port
            and destination MAC address,
            or it may choose to maintain multiple lldpRemEntries for
            the same local port and destination MAC address.

            The following procedure may be used to retrieve remote
            systems information updates from an LLDP agent:

               1. NMS polls all tables associated with remote systems
                  and keeps a local copy of the information retrieved.
                  NMS polls periodically the values of the following
                  objects:
```

```
                    a. lldpV2StatsRemTablesInserts
                    b. lldpV2StatsRemTablesDeletes
                    c. lldpV2StatsRemTablesDrops
                    d. lldpV2StatsRemTablesAgeouts
                    e. lldpV2StatsRxPortAgeoutsTotal for all ports.

                2. LLDP agent updates remote systems MIB objects, and
                   sends out notifications to a list of notification
                   destinations.

                3. NMS receives the notifications and compares the new
                   values of objects listed in step 1.

                   Periodically, NMS should poll the object
                   lldpV2StatsRemTablesLastChangeTime to find out if anything
                   has changed since the last poll. If something has
                   changed, NMS polls the objects listed in step 1 to
                   figure out what kind of changes occurred in the tables.

                   If value of lldpV2StatsRemTablesInserts has changed,
                   then NMS walks all tables by employing TimeFilter
                   with the last-polled time value. This request
                   returns new objects or objects whose values have been
                   updated since the last poll.

                   If value of lldpV2StatsRemTablesAgeouts has changed,
                   then NMS walks the lldpStatsRxPortAgeoutsTotal and
                   compares the new values with previously recorded ones.
                   For ports whose lldpStatsRxPortAgeoutsTotal value is
                   greater than the recorded value, NMS can
                   retrieve objects associated with those ports from
                   table(s) without employing a TimeFilter (which is
                   performed by specifying 0 for the TimeFilter).

                   lldpV2StatsRemTablesDeletes and lldpV2StatsRemTablesDrops
                   objects are provided for informational purposes."
    ::= { lldpV2RemoteSystemsData  1 }

lldpV2RemEntry OBJECT-TYPE
    SYNTAX      LldpV2RemEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Information about a particular physical network connection.
            Entries may be created and deleted in this table by the agent,
            if a physical topology discovery process is active.

            Rows in this table can only be created for MAC addresses
            that can validly be used in association with the type of
            interface concerned, as defined by Table 7-2.

            The contents of this table is persistent across
            re-initializations or re-boots."
    INDEX   {
           lldpV2RemTimeMark,
           lldpV2RemLocalIfIndex,
           lldpV2RemLocalDestMACAddress,
           lldpV2RemIndex
    }
```

```
    ::= { lldpV2RemTable 1 }

LldpV2RemEntry ::= SEQUENCE {
      lldpV2RemTimeMark              TimeFilter,
      lldpV2RemLocalIfIndex          InterfaceIndex,
      lldpV2RemLocalDestMACAddress   LldpV2DestAddressTableIndex,
      lldpV2RemIndex                 Unsigned32,
      lldpV2RemChassisIdSubtype      LldpV2ChassisIdSubtype,
      lldpV2RemChassisId             LldpV2ChassisId,
      lldpV2RemPortIdSubtype         LldpV2PortIdSubtype,
      lldpV2RemPortId                LldpV2PortId,
      lldpV2RemPortDesc              SnmpAdminString,
      lldpV2RemSysName               SnmpAdminString,
      lldpV2RemSysDesc               SnmpAdminString,
      lldpV2RemSysCapSupported       LldpV2SystemCapabilitiesMap,
      lldpV2RemSysCapEnabled         LldpV2SystemCapabilitiesMap,
      lldpV2RemRemoteChanges        TruthValue,
      lldpV2RemTooManyNeighbors      TruthValue
}

lldpV2RemTimeMark  OBJECT-TYPE
    SYNTAX      TimeFilter
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "A TimeFilter for this entry. See the TimeFilter textual
            convention in IETF RFC 4502 and
            http://www.ietf.org/IESG/Implementations/RFC2021-Implementation.txt
            to see how TimeFilter works."
    REFERENCE
            "IETF RFC 4502 section 6"
    ::= { lldpV2RemEntry 1 }


lldpV2RemLocalIfIndex   OBJECT-TYPE
    SYNTAX      InterfaceIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The interface index value used to identify the port
            associated with this entry. Its value is an index
            into the interfaces MIB

            The value of this object is used as an index to the
            lldpV2RemTable."
    ::= { lldpV2RemEntry 2 }

lldpV2RemLocalDestMACAddress   OBJECT-TYPE
    SYNTAX      LldpV2DestAddressTableIndex
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The index value used to identify the destination
            MAC address associated with this entry. Its value identifies
            the row in the lldpV2DestAddressTable where the MAC address
            can be found.

            The value of this object is used as an index to the
            lldpV2RemTable."
```

```
    ::= { lldpV2RemEntry 3 }


lldpV2RemIndex   OBJECT-TYPE
    SYNTAX      Unsigned32(1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This object represents an arbitrary local integer value used
            by this agent to identify a particular connection instance,
            unique only for the indicated remote system.

            An agent is encouraged to assign monotonically increasing
            index values to new entries, starting with one, after each
            reboot. It is considered unlikely that the lldpRemIndex
            can wrap between reboots."
    ::= { lldpV2RemEntry 4 }

lldpV2RemChassisIdSubtype   OBJECT-TYPE
    SYNTAX      LldpV2ChassisIdSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of encoding used to identify the chassis associated
            with the remote system."
    REFERENCE
            "8.5.2.2"
    ::= { lldpV2RemEntry 5 }

lldpV2RemChassisId   OBJECT-TYPE
    SYNTAX      LldpV2ChassisId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the chassis component
            associated with the remote system."
    REFERENCE
            "8.5.2.3"
    ::= { lldpV2RemEntry 6 }

lldpV2RemPortIdSubtype   OBJECT-TYPE
    SYNTAX      LldpV2PortIdSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The type of port identifier encoding used in the associated
            'lldpRemPortId' object."
    REFERENCE
            "8.5.3.2"
    ::= { lldpV2RemEntry 7 }

lldpV2RemPortId   OBJECT-TYPE
    SYNTAX      LldpV2PortId
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the port component
            associated with the remote system."
    REFERENCE
```

```
            "8.5.3.3"
    ::= { lldpV2RemEntry 8 }

lldpV2RemPortDesc   OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the description of
            the given port associated with the remote system."
    REFERENCE
            "8.5.5.2"
    ::= { lldpV2RemEntry 9 }

lldpV2RemSysName   OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the system name of the
            remote system."
    REFERENCE
            "8.5.6.2"
    ::= { lldpV2RemEntry 10 }

lldpV2RemSysDesc   OBJECT-TYPE
    SYNTAX      SnmpAdminString (SIZE(0..255))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the system description
            of the remote system."
    REFERENCE
            "8.5.7.2"
    ::= { lldpV2RemEntry 11 }

lldpV2RemSysCapSupported  OBJECT-TYPE
    SYNTAX      LldpV2SystemCapabilitiesMap
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The bitmap value used to identify which system capabilities
            are supported on the remote system."
    REFERENCE
            "8.5.8.1"
    ::= { lldpV2RemEntry 12 }

lldpV2RemSysCapEnabled   OBJECT-TYPE
    SYNTAX      LldpV2SystemCapabilitiesMap
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The bitmap value used to identify which system capabilities
            are enabled on the remote system."
    REFERENCE
            "8.5.8.2"
    ::= { lldpV2RemEntry 13 }

lldpV2RemRemoteChanges   OBJECT-TYPE
```

```
    SYNTAX      TruthValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Indicates that there are changes in the remote systems
            MIB, as determined by the variable remoteChanges."
    REFERENCE
            "9.2.5.11"
    ::= { lldpV2RemEntry 14 }

lldpV2RemTooManyNeighbors   OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "Indicates that there are too many neighbors
             as determined by the variable tooManyNeighbors."
    REFERENCE
            "9.2.5.15"
    ::= { lldpV2RemEntry 15 }

--
-- lldpV2RemManAddrTable : Management addresses of the remote system
-- Version 2 includes additional index values for ifIndex and
-- destination MAC address.
--

lldpV2RemManAddrTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2RemManAddrEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one or more rows per management address
            information on the remote system learned on a particular port
            contained in the local chassis known to this agent."
    ::= { lldpV2RemoteSystemsData 2 }

lldpV2RemManAddrEntry OBJECT-TYPE
    SYNTAX      LldpV2RemManAddrEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Management address information about a particular chassis
            component. There may be multiple management addresses
            configured on the remote system identified by a particular
            lldpRemIndex whose information is received on
            an interface of the local system and a given destination
            MAC address. Each management
            address should have distinct 'management address
            type' (lldpRemManAddrSubtype) and 'management address'
            (lldpRemManAddr).

            Entries may be created and deleted in this table by the
            agent.
            Since a variable length octetstring is used as an index
            in a table, theaddress length is encoded as part of the OID
            (as per IETF RFC 2578)."
    INDEX   { lldpV2RemTimeMark,
              lldpV2RemLocalIfIndex,
```

```
              lldpV2RemLocalDestMACAddress,
              lldpV2RemIndex,
              lldpV2RemManAddrSubtype,
              lldpV2RemManAddr
 }
    ::= { lldpV2RemManAddrTable 1 }


LldpV2RemManAddrEntry ::= SEQUENCE {
      lldpV2RemManAddrSubtype      AddressFamilyNumbers,
      lldpV2RemManAddr             LldpV2ManAddress,
      lldpV2RemManAddrIfSubtype    LldpV2ManAddrIfSubtype,
      lldpV2RemManAddrIfId         Unsigned32,
      lldpV2RemManAddrOID          OBJECT IDENTIFIER
}

lldpV2RemManAddrSubtype  OBJECT-TYPE
    SYNTAX      AddressFamilyNumbers
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The type of management address identifier encoding used in
            the associated 'lldpRemManagmentAddr' object.

            It should be noted that only a subset of the possible
            address encodings enumerated in AddressFamilyNumbers
            are appropriate for use as a LLDP management
            address, either because some are just not apliccable or
            because the maximum size of a LldpV2ManAddress octet string
            would prevent the use of some address identifier encodings."
    REFERENCE
            "8.5.9.3"
    ::= { lldpV2RemManAddrEntry 1 }

lldpV2RemManAddr  OBJECT-TYPE
    SYNTAX      LldpV2ManAddress
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The string value used to identify the management address
            component associated with the remote system. The purpose
            of this address is to contact the management entity."
    REFERENCE
            "8.5.9.4"
    ::= { lldpV2RemManAddrEntry 2 }

lldpV2RemManAddrIfSubtype  OBJECT-TYPE
    SYNTAX      LldpV2ManAddrIfSubtype
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The enumeration value that identifies the interface numbering
            method used for defining the interface number, associated
            with the remote system."
    REFERENCE
            "8.5.9.5"
    ::= { lldpV2RemManAddrEntry 3 }

lldpV2RemManAddrIfId  OBJECT-TYPE
```

```
    SYNTAX      Unsigned32
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The integer value used to identify the interface number
            regarding the management address component associated with
            the remote system. The value depends upon the value of the
            lldpV2RemManAddrIfSubtype for the table row."
    REFERENCE
            "8.5.9.6"
    ::= { lldpV2RemManAddrEntry 4 }

lldpV2RemManAddrOID  OBJECT-TYPE
    SYNTAX      OBJECT IDENTIFIER
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The OID value used to identify the type of hardware component
            or protocol entity associated with the management address
            advertised by the remote system agent."
    REFERENCE
            "8.5.9.8"
    ::= { lldpV2RemManAddrEntry 5 }


--
-- lldpV2RemUnknownTLVTable : Unrecognized TLV information
-- This version has additional indexes for
-- ifIndex and destination MAC address
--

lldpV2RemUnknownTLVTable  OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2RemUnknownTLVEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains information about an incoming TLV that
            is not recognized by the receiving LLDP agent. The TLV may
            be from a later version of the basic management set.

            This table should only contain TLVs that are found in
            a single LLDP frame. Entries in this table, associated
            with an MAC service access point (MSAP, the access point
            for MAC services provided to the LCC sublayer, defined
            in IEEE Standards Dictionary Online, which is also
            identified with a particular lldpRemLocalPortNum,
            lldpRemIndex pair) are overwritten with most recently
            received unrecognized TLV from the same MSAP, or they
            naturally age out when the rxInfoTTL timer
            (associated with the MSAP) expires."
    REFERENCE
            "9.2.7.7.1"
    ::= { lldpV2RemoteSystemsData 3 }

lldpV2RemUnknownTLVEntry OBJECT-TYPE
    SYNTAX      LldpV2RemUnknownTLVEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
```

```
            "Information about an unrecognized TLV received from a
            physical network connection. Entries may be created and
            deleted in this table by the agent, if a physical topology
            discovery process is active."
    INDEX   {
           lldpV2RemTimeMark,
           lldpV2RemLocalIfIndex,
           lldpV2RemLocalDestMACAddress,
           lldpV2RemIndex,
           lldpV2RemUnknownTLVType
    }
    ::= { lldpV2RemUnknownTLVTable 1 }

LldpV2RemUnknownTLVEntry ::= SEQUENCE {
      lldpV2RemUnknownTLVType      Unsigned32,
      lldpV2RemUnknownTLVInfo      OCTET STRING
}

lldpV2RemUnknownTLVType OBJECT-TYPE
    SYNTAX      Unsigned32(9..126)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This object represents the value extracted from the type
            field of the TLV."
    REFERENCE
            "9.2.7.7.1"
    ::= { lldpV2RemUnknownTLVEntry 1 }

lldpV2RemUnknownTLVInfo OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..511))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object represents the value extracted from the value
            field of the TLV."
    REFERENCE
            "9.2.7.7.1"
    ::= { lldpV2RemUnknownTLVEntry 2 }

-- ------------------------------------------------------------------------
-- Remote Systems Extension Table - Organizationally Defined Information
-- ------------------------------------------------------------------------
--
-- lldpV2RemOrgDefInfoTable - indexed by ifIndex and destination
-- MAC address.
--

lldpV2RemOrgDefInfoTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2RemOrgDefInfoEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one or more rows per physical network
            connection which advertises the organizationally defined
            information.

            Note that this table contains one or more rows of
            organizationally defined information that is not recognized
```

```
            by the local agent.

            If the local system is capable of recognizing any
            organizationally defined information, appropriate extension
            MIBs from the organization should be used for information
            retrieval."
    ::= { lldpV2RemoteSystemsData 4 }

lldpV2RemOrgDefInfoEntry OBJECT-TYPE
    SYNTAX      LldpV2RemOrgDefInfoEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "Information about the unrecognized organizationally
            defined information advertised by the remote system.
            The lldpRemTimeMark, lldpRemLocalPortNum, lldpRemIndex,
            lldpRemOrgDefInfoOUI, lldpRemOrgDefInfoSubtype, and
            lldpRemOrgDefInfoIndex are indexes to this table. If there is
            an lldpRemOrgDefInfoEntry associated with a particular remote
            system identified by the lldpRemLocalPortNum and lldpRemIndex,
            then there is an lldpRemEntry associated with the same
            instance (i.e., using same indexes.) When the lldpRemEntry
            for the same index is removed from the lldpRemTable, the
            associated lldpRemOrgDefInfoEntry is removed from
            the lldpRemOrgDefInfoTable.

            Entries may be created and deleted in this table by the
            agent."
    INDEX   { lldpV2RemTimeMark,
              lldpV2RemLocalIfIndex,
              lldpV2RemLocalDestMACAddress,
              lldpV2RemIndex,
              lldpV2RemOrgDefInfoOUI,
              lldpV2RemOrgDefInfoSubtype,
              lldpV2RemOrgDefInfoIndex }
    ::= { lldpV2RemOrgDefInfoTable 1 }

LldpV2RemOrgDefInfoEntry ::= SEQUENCE {
      lldpV2RemOrgDefInfoOUI         OCTET STRING,
      lldpV2RemOrgDefInfoSubtype     Unsigned32,
      lldpV2RemOrgDefInfoIndex       Unsigned32,
      lldpV2RemOrgDefInfo            OCTET STRING
}

lldpV2RemOrgDefInfoOUI  OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(3))
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The Organizationally Unique Identifier (OUI), as defined
            in IEEE Std 802, is a 24 bit (three octets) globally
            unique assigned number referenced by various standards,
            of the information received from the remote system."
    REFERENCE
            "8.6.1.3"
    ::= { lldpV2RemOrgDefInfoEntry 1 }

lldpV2RemOrgDefInfoSubtype  OBJECT-TYPE
    SYNTAX      Unsigned32(1..255)
```

```
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "The integer value used to identify the subtype of the
            organizationally defined information received from the
            remote system.

            The subtype value is required to identify different instances
            of organizationally defined information that could not be
            retrieved without a unique identifier that indicates the
            particular type of information contained in the information
            string."
    REFERENCE
            "8.6.1.4"
    ::= { lldpV2RemOrgDefInfoEntry 2 }

lldpV2RemOrgDefInfoIndex  OBJECT-TYPE
    SYNTAX      Unsigned32(1..2147483647)
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This object represents an arbitrary local integer value
            used by this agent to identify a particular unrecognized
            organizationally defined information instance, unique only
            for the lldpRemOrgDefInfoOUI and lldpRemOrgDefInfoSubtype
            from the same remote system.

            An agent is encouraged to assign monotonically increasing
            index values to new entries, starting with one, after each
            reboot. It is considered unlikely that the
            lldpRemOrgDefInfoIndex can wrap between reboots."
    ::= { lldpV2RemOrgDefInfoEntry 3 }

lldpV2RemOrgDefInfo  OBJECT-TYPE
    SYNTAX      OCTET STRING(SIZE(0..507))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "The string value used to identify the organizationally
            defined information of the remote system. The encoding for
            this object should be as defined for SnmpAdminString TC."
    REFERENCE
            "8.6.1.5"
    ::= { lldpV2RemOrgDefInfoEntry 4 }


--
-- ********************************************************
--
--        L L D P   M I B   N O T I F I C A T I O N S
--
-- ********************************************************
--

lldpV2NotificationPrefix OBJECT IDENTIFIER ::= { lldpV2Notifications 0 }

lldpV2RemTablesChange NOTIFICATION-TYPE
    OBJECTS {
        lldpV2StatsRemTablesInserts,
```

```
        lldpV2StatsRemTablesDeletes,
        lldpV2StatsRemTablesDrops,
        lldpV2StatsRemTablesAgeouts
    }
    STATUS        current
    DESCRIPTION
            "A lldpV2RemTablesChange notification is sent when the value
            of lldpV2StatsRemTablesLastChangeTime changes. It can be
            utilized by an NMS to trigger LLDP remote systems table
            maintenance polls.

            Note that transmission of lldpV2RemTablesChange
            notifications are throttled by the agent, as specified by the
            'lldpV2NotificationInterval' object."
   ::= { lldpV2NotificationPrefix 1 }


--
-- ********************************************************
--
--           L L D P   M I B   C O N F O R M A N C E
--
-- ********************************************************
--

lldpV2Compliances OBJECT IDENTIFIER ::= { lldpV2Conformance 1 }
lldpV2Groups      OBJECT IDENTIFIER ::= { lldpV2Conformance 2 }

-- compliance statements

lldpV2TxRxCompliance MODULE-COMPLIANCE
                        --V2 to add ifGeneralInformationGroup
                        --and support re-indexed tables
    STATUS  current
    DESCRIPTION
            "A compliance statement for all SNMP entities that
            implement the LLDP MIB as either a transmitter or
            a receiver of LLDPDUs.

            This version defines compliance requirements for
            V2 of the LLDP MIB module."
    MODULE  -- this module
        MANDATORY-GROUPS { lldpV2ConfigGroup,
                           ifGeneralInformationGroup
        }

    ::= { lldpV2Compliances 1 }

lldpV2TxCompliance MODULE-COMPLIANCE
                    --V2 requirements for transmitters of LLDPDUs
                       --and support re-indexed tables
    STATUS  current
    DESCRIPTION
            "A compliance statement for SNMP entities that implement
            the LLDP MIB and have the capability of transmitting
            LLDP frames.

            This version defines compliance requirements for
            V2 of the LLDP MIB module."
```

```
    MODULE  -- this module
        MANDATORY-GROUPS { lldpV2ConfigTxGroup,
                           lldpV2StatsTxGroup,
                           lldpV2LocSysGroup
        }

    ::= { lldpV2Compliances 2 }

lldpV2RxCompliance MODULE-COMPLIANCE
                      --V2 requirements for receivers of LLDPDUs
                      --and support re-indexed tables
    STATUS  current
    DESCRIPTION
            "The compliance statement for SNMP entities that implement
            the LLDP MIB and have the capability of receiving
            LLDP frames.

            This version defines compliance requirements for
            V2 of the LLDP MIB module."
    MODULE  -- this module
        MANDATORY-GROUPS { lldpV2ConfigRxGroup,
                           lldpV2StatsRxGroup,
                           lldpV2RemSysGroup,
                           lldpV2NotificationsGroup
        }

     ::= { lldpV2Compliances 3 }

-- MIB groupings


lldpV2ConfigGroup    OBJECT-GROUP
    OBJECTS {
        lldpV2PortConfigAdminStatusV2
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to configure the
            LLDP implementation behavior."
    ::= { lldpV2Groups 1 }


lldpV2ConfigRxGroup    OBJECT-GROUP
    OBJECTS {
        lldpV2NotificationInterval,
        lldpV2PortConfigNotificationEnableV2
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to configure the
            LLDP reception implementation behavior."
    ::= { lldpV2Groups 2 }


lldpV2ConfigTxGroup    OBJECT-GROUP
    OBJECTS {
        lldpV2MessageTxInterval,
        lldpV2MessageTxHoldMultiplier,
        lldpV2ReinitDelay,
```

```
        lldpV2PortConfigTLVsTxEnableV2,
        lldpV2ManAddrConfigTxEnable,
        lldpV2ManAddrConfigRowStatus,
        lldpV2TxCreditMax,
        lldpV2MessageFastTx,
        lldpV2TxFastInit,
        lldpV2DestMacAddress,
        lldpV2PortMessageTxInterval,
        lldpV2PortMessageTxHoldMultiplier,
        lldpV2PortReinitDelay,
        lldpV2PortNotificationInterval,
        lldpV2PortTxCreditMax,
        lldpV2PortMessageFastTx,
        lldpV2PortTxFastInit
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to configure the
            LLDP transmission implementation behavior."
    ::= { lldpV2Groups 3 }


lldpV2StatsRxGroup    OBJECT-GROUP
    OBJECTS {
        lldpV2StatsRemTablesLastChangeTime,
        lldpV2StatsRemTablesInserts,
        lldpV2StatsRemTablesDeletes,
        lldpV2StatsRemTablesDrops,
        lldpV2StatsRemTablesAgeouts,
        lldpV2StatsRxPortFramesDiscardedTotal,
        lldpV2StatsRxPortFramesErrors,
        lldpV2StatsRxPortFramesTotal,
        lldpV2StatsRxPortTLVsDiscardedTotal,
        lldpV2StatsRxPortTLVsUnrecognizedTotal,
        lldpV2StatsRxPortAgeoutsTotal
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to represent LLDP
            reception statistics."
    ::= { lldpV2Groups 4 }


lldpV2StatsTxGroup    OBJECT-GROUP
    OBJECTS {
        lldpV2StatsTxPortFramesTotal,
        lldpV2StatsTxLLDPDULengthErrors
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to represent LLDP
            transmission statistics."
    ::= { lldpV2Groups 5 }


lldpV2LocSysGroup  OBJECT-GROUP
    OBJECTS {
        lldpV2LocChassisIdSubtype,
        lldpV2LocChassisId,
```

```
        lldpV2LocPortIdSubtype,
        lldpV2LocPortId,
        lldpV2LocPortDesc,
        lldpV2LocSysDesc,
        lldpV2LocSysName,
        lldpV2LocSysCapSupported,
        lldpV2LocSysCapEnabled,
        lldpV2LocManAddrLen,
        lldpV2LocManAddrIfSubtype,
        lldpV2LocManAddrIfId,
        lldpV2LocManAddrOID
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to represent LLDP
            Local System Information."
    ::= { lldpV2Groups 6 }

lldpV2RemSysGroup  OBJECT-GROUP
    OBJECTS {
        lldpV2RemChassisIdSubtype,
        lldpV2RemChassisId,
        lldpV2RemPortIdSubtype,
        lldpV2RemPortId,
        lldpV2RemPortDesc,
        lldpV2RemSysName,
        lldpV2RemSysDesc,
        lldpV2RemSysCapSupported,
        lldpV2RemSysCapEnabled,
        lldpV2RemRemoteChanges,
        lldpV2RemTooManyNeighbors,
        lldpV2RemManAddrIfSubtype,
        lldpV2RemManAddrIfId,
        lldpV2RemManAddrOID,
        lldpV2RemUnknownTLVInfo,
        lldpV2RemOrgDefInfo
    }
    STATUS  current
    DESCRIPTION
            "The collection of objects that are used to represent
            LLDP Remote Systems Information. The objects represent the
            information associated with the basic TLV set. Please note
            that even if the agent does not implement some of the optional
            TLVs, it shall recognize all the optional TLV information
            that the remote system may advertise."
    ::= { lldpV2Groups 7 }

lldpV2NotificationsGroup  NOTIFICATION-GROUP
    NOTIFICATIONS {
        lldpV2RemTablesChange
    }
    STATUS  current
    DESCRIPTION
            "The collection of notifications used to indicate LLDP MIB
            data consistency and general status information."
    ::= { lldpV2Groups 8 }

END
```

Annex A

(normative)

PICS proforma[13]

A.1 Introduction

The supplier of a protocol implementation that is claimed to conform to this standard shall complete the following Protocol Implementation Conformance Statement (PICS) proforma.

A completed PICS proforma is the PICS for the implementation in question. The PICS is a statement of which capabilities and options of the protocol have been implemented. The PICS can have a number of uses, including use

a) By the protocol implementers, as a checklist to reduce the risk of failure to conform to the standard through oversight.
b) By the supplier and acquirer—or potential acquirer—of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma.
c) By the user—or potential user—of the implementation, as a basis for initially checking the possibility of interworking with another implementation (note that although interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICS).
d) By a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A.2 Abbreviations and special symbols

A.2.1 Status symbols

M Mandatory
O Optional
O.n Optional, but support of at least one of the group of options labeled by the same numeral *n* is required
X Prohibited
pred: Conditional-item symbol, including predicate identification: See B.3.4
¬ Logical negation, applied to a conditional item's predicate

A.2.2 General abbreviations

N/A Not applicable
PICS Protocol Implementation Conformance Statement

[13] *Copyright release for PICS proformas:* Users of this standard may freely reproduce the PICS proforma in this annex so that it can be used for its intended purpose and may further publish the completed PICS.

A.3 Instructions for completing the PICS proforma

A.3.1 General structure of the PICS proforma

The first part of the PICS proforma, implementation identification and protocol summary, is to be completed as indicated with the information necessary to identify fully both the supplier and the implementation.

The main part of the PICS proforma is a fixed-format questionnaire, divided into several subclauses, each containing a number of individual items. Answers to the questionnaire items are to be provided in the right-most column, either by simply marking an answer to indicate a restricted choice (usually Yes or No), or by entering a value or a set or range of values. (Note that there are some items in which two or more choices from a set of possible answers can apply; all relevant choices are to be marked.)

Each item is identified by an item reference in the first column. The second column contains the question to be answered; the third column records the status of the item—whether support is mandatory, optional, or conditional; see also B.3.4. The fourth column contains the reference or references to the material that specifies the item in the main body of this standard, and the fifth column provides the space for the answers.

A supplier may also provide (or be required to provide) further information, categorized as either Additional Information or Exception Information. When present, each kind of further information is to be provided in a further subclause of items labeled *Ai* or *Xi,* respectively, for cross-referencing purposes, where *i* is any unambiguous identification for the item (e.g., simply a numeral). There are no other restrictions on its format and presentation.

A completed PICS proforma, including any Additional Information and Exception Information, is the Protocol Implementation Conformation Statement for the implementation in question.

NOTE—Where an implementation is capable of being configured in more than one way, a single PICS may be able to describe all such configurations. However, the supplier has the choice of providing more than one PICS, each covering some subset of the implementation's configuration capabilities, in case that makes for easier and clearer presentation of the information.

A.3.2 Additional information

Items of Additional Information allow a supplier to provide further information intended to assist the interpretation of the PICS. It is not intended or expected that a large quantity will be supplied, and a PICS can be considered complete without any such information. Examples might be an outline of the ways in which a (single) implementation can be set up to operate in a variety of environments and configurations, or information about aspects of the implementation that are outside the scope of this standard but that have a bearing on the answers to some items.

References to items of Additional Information may be entered next to any answer in the questionnaire and may be included in items of Exception Information.

A.3.3 Exception information

It may occasionally happen that a supplier will wish to answer an item with mandatory status (after any conditions have been applied) in a way that conflicts with the indicated requirement. No preprinted answer will be found in the Support column for this: instead, the supplier shall write the missing answer into the Support column, together with an *Xi* reference to an item of Exception Information, and shall provide the appropriate rationale in the Exception item.

An implementation for which an Exception item is required in this way does not conform to this standard.

NOTE—A possible reason for the situation described above is that a defect in this standard has been reported, a correction for which is expected to change the requirement not met by the implementation.

A.3.4 Conditional status

A.3.4.1 Conditional items

The PICS proforma contains a number of conditional items. These are items for which both the applicability of the item itself, and its status if it does apply—mandatory or optional—are dependent upon whether or not certain other items are supported.

Where a group of items is subject to the same condition for applicability, a separate preliminary question about the condition appears at the head of the group, with an instruction to skip to a later point in the questionnaire if the "Not Applicable" answer is selected. Otherwise, individual conditional items are indicated by a conditional symbol in the Status column.

A conditional symbol is of the form "pred: S," where pred is a predicate as described in A.3.4.2, and S is a status symbol, M or O.

If the value of the predicate is TRUE (see A.3.4.2), the conditional item is applicable, and its status is indicated by the status symbol following the predicate: the answer column is to be marked in the usual way. If the value of the predicate is FALSE, the "Not Applicable" (N/A)[14] answer is to be marked.

A.3.4.2 Predicates

A predicate is one of the following:

a) An item-reference for an item in the PICS proforma: The value of the predicate is TRUE if the item is marked as supported, and is FALSE otherwise.
b) A predicate-name, for a predicate defined as a Boolean expression constructed by combining item-references using the Boolean operator OR: The value of the predicate is TRUE if one or more of the items is marked as supported.
c) A predicate-name, for a predicate defined as a Boolean expression constructed by combining item-references using the Boolean operator AND: The value of the predicate is TRUE if all of the items are marked as supported.
d) The logical negation symbol "¬" prefixed to an item-reference or predicate-name: The value of the predicate is TRUE if the value of the predicate formed by omitting the "¬" symbol is FALSE, and vice versa.

Each item whose reference is used in a predicate or predicate definition, or in a preliminary question for grouped conditional items, is indicated by an asterisk[15] in the Item column.

[14](N/A) is currently not used in this PICS.
[15]Asterisks are currently not used in this PICS.

PICS proforma for IEEE Std 802.1AB-2016

A.3.5 Implementation identification

Supplier	
Contact point for queries about the PICS	
Implementation Name(s) and Version(s)	
Other information necessary for full identification—e.g., name(s) and version(s) of machines and/or operating system names	
NOTE 1—Only the first three items are required for all implementations; other information may be completed as appropriate in meeting the requirement for full identification. NOTE 2—The terms Name and Version should be interpreted appropriately to correspond with a supplier's terminology (e.g., Type, Series, Model).	

A.3.6 Protocol summary, IEEE Std 802.1AB-2016

Identification of protocol specification	IEEE Std 802.1AB-2016, IEEE Standard for Local and metropolitan area networks—Station and Media Access Control Connectivity Discovery
Identification of amendments and corrigenda to the PICS proforma that have been completed as part of the PICS	Amd. : Corr. : Amd. : Corr. :
Have any Exception items been required? (See B.3.3: The answer Yes means that the implementation does not conform to IEEE Std 802.1AB-2016)	No [ ] Yes [ ]

Date of Statement	

A.4 Major capabilities and options

Item	Feature	Status	References	Support
cntrlport	Are LLDP exchanges supported through a controlled port if port access is controlled by IEEE Std 802.1X?	M	5.3, 6	Yes [ ] N/A [ ]
uncntrlport	Are LLDP exchanges supported through the uncontrolled port if port access is controlled by IEEE Std 802.1X?	O	5.4, 6	Yes [ ] No [ ] N/A [ ]
addr	Are LLDP addressing and LLDP EtherType encoding in conformance with the defined requirements?			
	All systems:			
	DA = Any group MAC address	O	7.1	Yes [ ] No [ ]
	DA = Any individual MAC address	O	7.1	Yes [ ] No [ ]
	SA = station MAC address	M	7.2	Yes [ ]
	LLDP EtherType encoding	M	7.3	Yes [ ]
	C-VLAN Bridge:			
	DA = Nearest bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest non-TPMR bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest customer bridge address	M	7.1	Yes [ ] N/A [ ]
	S-VLAN Bridge:			
	DA = Nearest bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest non-TPMR bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest customer bridge address	X	7.1	No [ ] N/A [ ]
	TPMR Bridge:			
	DA = Nearest bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest non-TPMR bridge address	X	7.1	No [ ] N/A [ ]
	DA = Nearest customer bridge address	X	7.1	No [ ] N/A [ ]
	End station:			
	DA = Nearest bridge address	M	7.1	Yes [ ] N/A [ ]
	DA = Nearest non-TPMR bridge address	O	7.1	Yes [ ] No [ ] N/A [ ]
	DA = Nearest customer bridge address	O	7.1	Yes [ ] No [ ] N/A [ ]
lldpdu	Is the LLDPDU encapsulation in conformance with the TLV order specified by the LLDPDU format?	M	7.3	Yes [ ]
tlvfmt	Is the basic TLV capability implemented?	M	8.4	Yes [ ]
basictlv	Is each TLV in the basic management set implemented?			
	End Of LLDPDU TLV	O	8.5.1	Yes [ ] No [ ]
	Chassis ID TLV	M	8.5.2	Yes [ ]
	Port ID TLV	M	8.5.3	Yes [ ]
	Time To Live TLV	M	8.5.4	Yes [ ]
	Port Description TLV	M	8.5.5	Yes [ ]
	System Name TLV	M	8.5.6	Yes [ ]
	System Description TLV	M	8.5.7	Yes [ ]
	System Capabilities TLV	M	8.5.8	Yes [ ]
	Management Address TLV	M	8.5.9	Yes [ ]
xtlvfmt	Is the Organizationally Specific TLV capability implemented?	M	8.6	Yes [ ]

A.4 Major capabilities and options

Item	Feature	Status	References	Support
	Which of the following operational modes are implemented?			
optxrx	Transmit and receive	O.1	6.1	Yes [ ] No [ ]
optx	Transmit only	O.1	6.1	Yes [ ] No [ ]
oprx	Receive only	O.1	6.1	Yes [ ] No [ ]
txmode	Is the transmit mode in conformance with all operational specifications indicated for the Tx mode in Table 9-1?	optxrx OR optx: M	Clause 9	Yes [ ] N/A [ ]
rxmode	Is the receive module in conformance with all operational specifications indicated for the Tx mode in Table 9-1?	optxrx OR oprx: M	Clause 9	Yes [ ] N/A [ ]
	Which type of data store/retrieval is implemented?			
mib	SNMP MIB is supported	O.2	11.5, 5.3	Yes [ ] No [ ]
nomib	SNMP MIB is not supported	O.2	10.1, 5.3	Yes [ ] No [ ]
snmpmib	Is the MIB module in conformance with the MIB sections indicated in Table 11-1 for the operating mode being implemented?	mib:M	11.5, 5.3	Yes [ ] N/A [ ]
snmpsupport	Which of the transport mappings defined by IETF RFC 3417 or IETF RFC 4789 is used to support SNMP?			
	IETF RFC 3417	mib:O.3	5.3, 5.4	Yes [ ] No [ ] N/A [ ]
	IETF RFC 4789	mib:O.3	5.3, 5.4	Yes [ ] No [ ] N/A [ ]
equivstor	If the SNMP is not supported, is functionally equivalent storage and retrieval capability specified in Clause 8, Clause 9, Clause10 provided for the operating mode being implemented?	nomib:M	10.1	Yes [ ] N/A [ ]

Annex B

(normative)

PTOPO MIB update

The PTOPO MIB should be updated according to the following rules:

a) If any objects in the LLDP remote systems MIB age out, the equivalent objects in the PTOPO MIB should be deleted.

b) If the TTL value in the Time To Live TLV is zero, then the port is being shutdown and the PTOPO MIB objects associated with the MSAP identifier should be deleted.

c) If the TTL field is non-zero, then the appropriate ptopoRemEntry is found or created, based on the data elements included in the LLDP frame. If the indicated entry is dynamic (i.e., ptopoConnIsStatic is FALSE), then the current sysUpTime value is stored in the ptopoConnLastVerifyTime field for the entry.

d) If a ptopoRemEntry was added then the ptopoConnTabInserts counter is incremented.

e) If any ptopoRemEntry was added or deleted, or if information other than the ptopoRemLastVerifyTime changed for any entry due to the processing of this LLDP frame, the ptopoLastChangeTime object is set with the current sysUpTime, and a ptopoConfigChange trap event is generated. (See the PTOPO MIB for information on ptopoConfigChange trap generation.)

Annex C

(informative)

Example LLDP transmission frame formats

The LLDP MAC frame format is based on the particular transmission protocol. The following example formats illustrate the indicated LLDP EtherType encoding method.

C.1 Direct-encoded LLDP frame format

The IEEE 802.3 LLDP frame format is illustrated in Figure C.1.

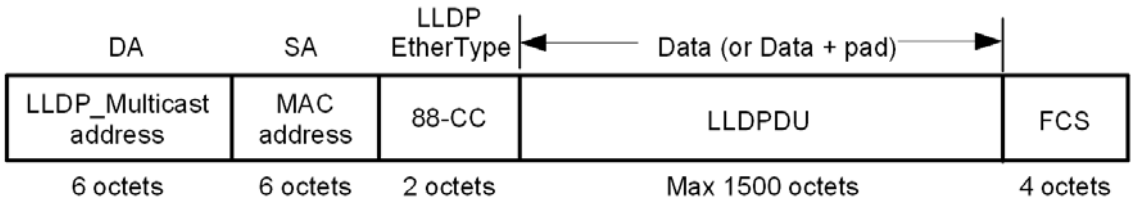

Figure C.1—IEEE 802.3 LLDP frame format

NOTE—The illustration shows the simplest form of an LLDP frame on an IEEE 802.3 medium; i.e., where the frame has had no IEEE 802.1Q tag header, or IEEE 802.1AE™ security tag, or any other form of encapsulation applied to it.

C.2 SNAP-encoded LLDP frame format

The IEEE 802.11™ frame format is illustrated in Figure C.2, for the case where the value of To DS and From DS in the Frame Control field are both zero.

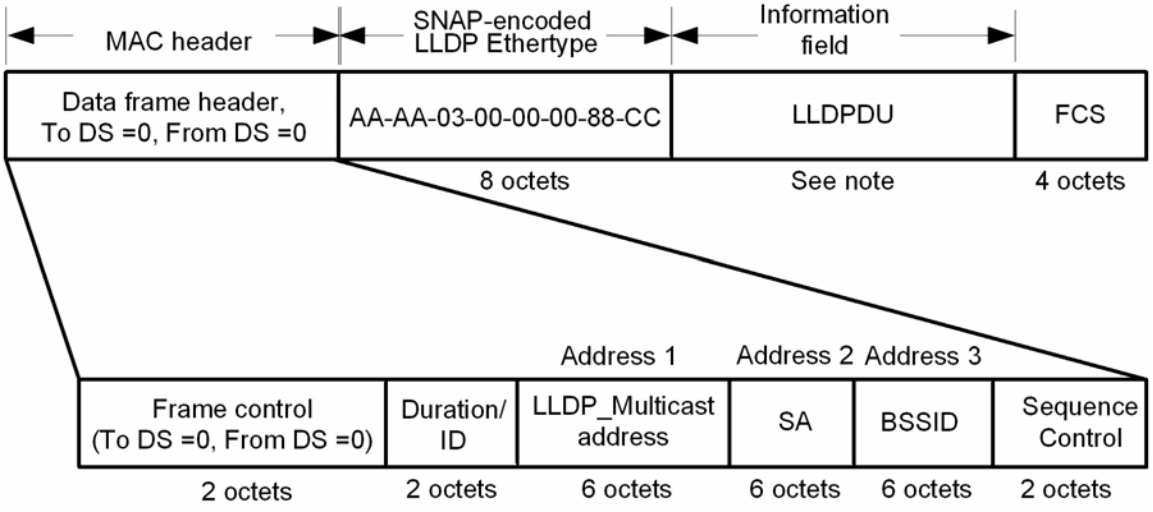

Figure C.2—IEEE 802.11 LLDP frame format

NOTE—The illustration shows the simplest form of an LLDP frame on an IEEE 802.11 medium; i.e., where the frame has had no IEEE 802.1Q tag header, or IEEE 802.1AE security tag, or any other form of encapsulation applied to it.

Annex D

(informative)

Bibliography

[B1] IEEE Std 802.1aj-2009, Standard for Local and Metropolitan Area Networks—Virtual Bridged Local Area Networks—Amendment: Two-port Media Access Control (MAC) Relay.[16]

[B2] IEEE Std 802.11, Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.

[B3] IETF RFC 2578 (STD 58), Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2), McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.[17]

[B4] IETF RFC 2579 (STD 58), Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2), McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.

[B5] IETF RFC 2580 (STD 58), Conformance Statements for SMIv2, McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M., Waldbusser, S., April 1999.

[B6] IETF RFC 1812, Requirements for IP Version 4 Routers, June 1995.

[B7] IETF RFC 2108, Definitions of Managed Objects for IEEE 802.3 Repeater Devices using SMIv2, February 1997.

[B8] IETF RFC 2670, Radio Frequency (RF) Interface Management Information Base for MCNS/DOCSIS compliant RF interfaces, August 1999.

[B9] IETF RFC 2922, Physical Topology MIB, Bierman, A., and Jones, K., November 1998.

[B10] IETF RFC 4293, Management Information Base for the Internet Protocol (IP), April 2006.

[B11] IETF RFC 4363, Definitions of Managed Objects for Bridges with Traffic Classes, Multicast Filtering, and Virtual LAN Extensions, January 2006.

[B12] IETF RFC 4546, Radio Frequency (RF) Interface Management Information Base for Data over Cable Service Interface Specifications (DOCSIS) 2.0 Compliant RF Interfaces, June 2006.

[16]IEEE publications are available from The Institute of Electrical and Electronic Engineers (http://standards.ieee.org).
[17]IETF documents (i.e., RFCs) are available for download at http://www.rfc-archive.org/.

Consensus

WE BUILD IT.

Connect with us on:

Facebook: https://www.facebook.com/ieeesa

Twitter: @ieeesa

LinkedIn: http://www.linkedin.com/groups/IEEESA-Official-IEEE-Standards-Association-1791118

IEEE-SA Standards Insight blog: http://standardsinsight.com

YouTube: IEEE-SA Channel

IEEE
standards.ieee.org
Phone: +1 732 981 0060 Fax: +1 732 562 1571