IEEE SA
STANDARDS ASSOCIATION

STANDARDS

IEEE Standard for Local and Metropolitan Area Networks—

Bridges and Bridged Networks

Amendment 37: Automatic Attachment to Provider Backbone Bridging (PBB) Services

IEEE Computer Society

Developed by the
LAN/MAN Standards Committee

IEEE Std 802.1Qcj™-2023
(Amendment to IEEE Std 802.1Q™-2018
as amended by IEEE Std 802.1Qcz™-2023
and IEEE Std 802.1Qcw™-2023)

IEEE

IEEE Std 802.1Qcj™-2023
(Amendment to IEEE Std 802.1Q™-2022
as amended by IEEE Std 802.1Qcz™-2023
and IEEE Std 802.1Qcw™-2023)

IEEE Standard for Local and Metropolitan Area Networks—

Bridges and Bridged Networks

Amendment 37: Automatic Attachment to Provider Backbone Bridging (PBB) Services

Developed by the

LAN/MAN Standards Committee
of the
IEEE Computer Society

Approved 21 September 2023

IEEE SA Standards Board

Abstract: This amendment to IEEE Std 802.1Q-2022 as amended by IEEE Std 802.1Qcz-2023 and IEEE Std 802.1Qcw-2023 specifies protocols, procedures, and management objects for auto attachment of network devices to Provider Backbone service instances using Type, Length, Value (TLVs) within the Link Layer Discovery Protocol (LLDP).

Keywords: AAB, AAD, AAP, amendment, Auto Attach, Auto Attach Backbone Edge Bridge, Auto Attach Device, Auto Attach Protocol, Bridged Network, IEEE 802.1Q™, IEEE 802.1Qcj™, LAN, local area network, MAC Bridge, metropolitan area network, MSTP, Multiple Spanning Tree Protocol, PBBN, PBN, Provider Backbone Bridged Network, Provider Bridged Network, Rapid Spanning Tree Protocol, RSTP, Shortest Path Bridging Protocol, SPB Protocol, Time-Sensitive Networking, TSN, Virtual Bridged Network, virtual LAN, VLAN Bridge

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

PDF: ISBN 979-8-8557-0268-2 STD26585
Print: ISBN 979-8-8557-0269-9 STDPD26585

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (https://standards.ieee.org/ipr/disclaimers.html), appear in all standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents."

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

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA 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 makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning this standard, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. IEEE Standards documents do not guarantee safety, security, health, or environmental protection, or guarantee against interference with or from other devices or networks. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. 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 their 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: THE NEED TO PROCURE 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 balloting 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 is 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, nor be relied upon 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 the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s).

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, 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 subcommittees of the IEEE SA Board of Governors 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 evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the IEEE SA myProject system.[1] An IEEE Account is needed to access the application.

Comments on standards should be submitted using the Contact Us form.[2]

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 constitute 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.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

---

[1] Available at: https://development.standards.ieee.org/myproject-web/public/view.html#landing.

[2] Available at: https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html.

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, neither IEEE nor its licensors waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, 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; https://www.copyright.com/. 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 10 years. When a document is more than 10 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 IEEE Xplore or contact IEEE.[3] For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE SA Website.[4] Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in IEEE Xplore. Users are encouraged to periodically check for errata.

Patents

IEEE standards are developed in compliance with the IEEE SA Patent Policy.[5]

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

[3] Available at: https://ieeexplore.ieee.org/browse/standards/collection/ieee.

[4] Available at: https://standards.ieee.org/standard/index.html.

[5] Available at: https://standards.ieee.org/about/sasb/patcom/materials.html.

filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at https://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.

IMPORTANT NOTICE

Technologies, application of technologies, and recommended procedures in various industries evolve over time. The IEEE standards development process allows participants to review developments in industries, technologies, and practices, and to determine what, if any, updates should be made to the IEEE standard. During this evolution, the technologies and recommendations in IEEE standards may be implemented in ways not foreseen during the standard's development. IEEE standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

Participants

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

Glenn Parsons, *Chair*

Jessy V. Rouyer, *Vice Chair*

János Farkas, *Chair, Time-Sensitive Networking Task Group*

Craig Gunther, *Vice Chair, Time-Sensitive Networking Task Group*

Paul Bottorff, *Editor*

Katsuyuki Akizuki
Konstantinos Alexandris
Venkat Arunarthi
Ralf Assmann
Huajie Bao
Rudy Belliardi
Jeremias Blendin
Christian Boiger
Radhakrishna Canchi
Feng Chen
Abhijit Choudhury
Paul Congdon
Rodney Cummings
Josef Dorr
Hesham Elbakoury
Anna Engelmann
Thomas Enzinger
Donald Fedyk
Norman Finn
Geoffrey Garner
Marina Gutiérrez
Stephen Haddock
Mark Hantel
Marc Holness
Daniel Hopf
Woojung Huh
Satoko Itaya
Yoshihiro Ito
Michael Karl
Stephan Kehrer
Marcel Kiessling
Gavin Lai
Yizhou Li
Joao Lopes
Lily Lyu
Christophe Mangin
Scott Mansfield
Olaf Mater
David McCall
Larry McMillan
Martin Mittelberger
Hiroki Nakano
Takumi Nomura
Dragan Obradovic
Donald R. Pannell
Dieter Proell
Karen Randall
Maximilian Riegel
Craig Rodine
Silvana Rodrigues
Atsushi Sato
Frank Schewe
Michael Seaman
Maik Seewald
Ramesh Sivakolundu
Johannes Specht
Nemanja Stamenic
Marius Stanica
Guenter Steindl
Karim Traore
Max Turner
Balazs Varga
Ganesh Venkatesan
Tongtong Wang
Karl Weber
Leon Wessels
Ludwig Winkel
Jordon Woods
Takahiro Yamaura
Nader Zein

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

Thomas Alexander
Amelia Andersdotter
Butch Anton
Christian Boiger
Paul Bottorff
William Byrd
Paul Cardinal
Jose Castro
Pin Chang
Aditya Chaudhuri
Paul Congdon
János Farkas
Donald Fedyk
Avraham Freedman
Chong Han
Marco Hernandez
Werner Hoelzl
Yasuhiro Hyakutake
Pranav Jha
Lokesh Kabra
Piotr Karocki
Stuart Kerry
Yongbum Kim
Hyeong Ho Lee
James Lepp
Greg Luri
Christophe Mangin
Scott Mansfield
Jonathon McLendon
Ronald Murias
Rajesh Murthy
Satoshi Obara
Glenn Parsons
Bansi Patel
Arumugam Paventhan
Clinton Powell
Dieter Proell
R. K. Rannow
Maximilian Riegel
Jessy V. Rouyer
Michael Seaman
Veselin Skendzic
Walter Struppler
Max Turner
John Vergis
James Weaver
Stephen Webb
Karl Weber
Scott Willy
Andreas Wolf
Yu Yuan
Oren Yuen
Qiyue Zou

When the IEEE SA Standards Board approved this standard on 21 September 2023, it had the following membership:

David J. Law, *Chair*
Ted Burse, *Vice Chair*
Gary Hoffman, *Past Chair*
Konstantinos Karachalios, *Secretary*

Sara R. Biyabani	Joseph S. Levy	Paul Nikolich
Doug Edwards	Howard Li	Annette D. Reilly
Ramy Ahmed Fathy	Johnny Daozhuang Lin	Robby Robson
Guido R. Hiertz	Gui Lin	Lei Wang
Yousef Kimiagar	Xiaohui Liu	F. Keith Waters
Joseph L. Koepfinger*	Kevin W. Lu	Karl Weber
Thomas Koshy	Daleep C. Mohla	Philip B. Winston
John D. Kulick	Andrew Myles	Don Wright

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 802.1Qcj™-2023, IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks—Amendment 37: Automatic Attachment to Provider Backbone Bridging (PBB) Services.

IEEE Std 802.1Qcj™-2023: Automatic Attachment to Provider Backbone Bridging (PBB) Services specifies protocols, procedures, and management objects for auto attachment of network devices to Provider Backbone service instances using Type, Length, Value (TLVs) within the Link Layer Discovery Protocol (LLDP).

This standard contains state-of-the-art material. The area covered by this standard is undergoing evolution. Revisions are anticipated within the next few years to clarify existing material, to correct possible errors, and to incorporate new related material. Information on the current revision state of this and other IEEE 802 standards may be obtained from:

Secretary, IEEE SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854-4141
USA

Contents

Figures

Tables

IEEE Standard for
Local and Metropolitan Area Networks—

Bridges and Bridged Networks

Amendment 37: Automatic Attachment to Provider Backbone Bridging (PBB) Services

(This amendment is based on IEEE Std 802.1Q™-2022 as amended by IEEE Std 802.1Qcz™-2023 and IEEE Std 802.1Qcw™-2023.)

NOTE—The editing instructions contained in this amendment define how to merge the material contained therein into the existing base standard and its amendments to form the comprehensive standard.

The editing instructions are shown in ***bold italics***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strikethrough~~ (to remove old material) and underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Deletions and insertions may require renumbering. If so, renumbering instructions are given in the editing instruction. ***Replace*** is used to make changes in figures or equations by removing the existing figure or equation and replacing it with a new one. Editing instructions, change markings, and this note will not be carried over into future editions because the changes will be incorporated into the base standard. [6]

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

1. Overview

1.3 Introduction

Insert the following text at the end of 1.3:

This standard specifies the protocols, procedures, and management objects for auto attachment of network devices to backbone service instances (BSIs) by adding an Auto Attach Protocol (AAP) over the Link Layer Discovery Protocol (LLDP). The AAP simplifies the deployment and administration of PBB networks by automatically coupling end stations or Bridges to BSIs.

To this end, it:

cz) Introduces the concepts of Auto Attach Protocol (AAP), Auto Attach System (AAS) on devices supporting Auto Attach; [7] non-PBB-based Auto Attach Devices (AADs) and PBB-based Auto Attach BEBs (AABs).

da) Describes the process of discovery and advertising of capabilities for AADs and AABs supporting the AAP acting as an AAS.

db) Describes the bindings between Backbone Service Instance Identifiers (I-SIDs) and VLAN Identifiers.

dc) Specifies IEEE 802.1 Organizationally Specific LLDP TLVs for discovery and capabilities advertising, and attachment of C-VLANs to backbone service instances.

dd) Specifies SNMP MIB support for PBBN Auto Attach.

de) Specifies YANG extensions to the LLDP YANG modules for the PBBN Auto Attach LLDP TLVs.

df) Specifies SNMP MIB extensions to the LLDP EVB extension MIB module for the PBBN Auto Attach LLDP TLVs.

[7] The term “Auto Attach” is used interchangeably with “PBBN Auto Attach” where it is clear from the context that “PBBN Auto Attach” is discussed.

3. Definitions

Insert the following definitions in the appropriate collating sequence, renumbering accordingly:

3.1 Auto Attach BEB (AAB) Bridge component: A Backbone Edge Bridge having at least one I-component that uses the AAP to respond to requests from AADs to couple LANs or VLANs to backbone service instances within a PBBN.

3.2 Auto Attach Device (AAD): An end station, MAC Bridge, C-VLAN Bridge, or S-VLAN Bridge that uses the AAP to request attachment to a backbone service instance within a PBBN.

3.3 Auto Attach Protocol (AAP): A protocol operating over LLDP used by an Auto Attach System (AAS) which provides auto attachment to a PBBN.

3.4 Auto Attach System (AAS): An Auto Attach Device (AAD) or Auto Attach BEB (AAB).

4. Abbreviations

Insert the following abbreviations into Clause 4, in the appropriate collating sequence:

AAB	Auto Attach BEB
AAD	Auto Attach Device
AAP	Auto Attach Protocol
AAS	Auto Attach System

5. Conformance

5.9 C-VLAN Bridge conformance

Insert 5.9.2 after 5.9.1 as follows:

5.9.2 C-VLAN Bridge requirements for PBBN Auto Attach (optional)

A C-VLAN Bridge that conforms to the provisions for PBBN Auto Attach (Clause 50) shall:

a) Support Link Layer Discovery Protocol (LLDP) transmit and receive mode (IEEE Std 802.1AB™).
b) Support the PBBN Auto Attach System TLV (D.2.17).
c) Support the PBBN Auto Attach Assignment TLV (D.2.18).
d) Support the AAD state machine (50.6).
e) Support management entities for PBBN Auto Attach as specified in 12.34.

A C-VLAN Bridge that conforms to the provisions for PBBN Auto Attach may:

f) Support SNMP MIB extensions for AAP (17.7.26).
g) Support SNMP MIB extensions for AAP LLDP TLVs (D.5.6).
h) Support YANG for AAP LLDP TLVs (D.6.6.7).

5.10 Provider Bridge conformance

Insert 5.10.3 after 5.10.2 as follows:

5.10.3 S-VLAN Bridge requirements for PBBN Auto Attach (optional)

An S-VLAN Bridge that conforms to the provisions for PBBN Auto Attach (Clause 50) shall:

a) Support Link Layer Discovery Protocol (LLDP) transmit and receive mode (IEEE Std 802.1AB).
b) Support the PBBN Auto Attach System TLV (D.2.17).
c) Support the PBBN Auto Attach Assignment TLV (D.2.18).
d) Support the AAD state machine (50.6).
e) Support management entities for PBBN Auto Attach as specified in 12.34.

An S-VLAN Bridge that conforms to the provisions for PBBN Auto Attach may:

f) Support SNMP MIB extensions for AAP (17.7.26).
g) Support SNMP MIB extensions for AAP LLDP TLVs (D.5.6).
h) Support YANG for AAP LLDP TLVs (D.6.6.7).

5.12 Backbone Edge Bridge (BEB) conformance

Insert 5.12.2 after 5.12.1 as follows:

5.12.2 Backbone Edge Bridge requirements for PBBN Auto Attach (optional)

An Auto Attach BEB (AAB) shall be a conformant Backbone Edge Bridge (BEB, 5.12) with at least one I-component (5.7) and shall have one or more C-VLAN components providing externally visible Customer Edge Ports (CEP) as specified in 50.2.

Each externally accessible AAB port that supports PBBN Auto Attach shall:

a) Support Link Layer Discovery Protocol (LLDP) transmit and receive mode (IEEE Std 802.1AB).
b) Support the PBBN Auto Attach System TLV (D.2.17).
c) Support the PBBN Auto Attach Assignment TLV (D.2.18).
d) Support the C-tagging service interface (50.2.2) and S-tagging service interface (50.2.1).
e) Support the AAB state machine (50.7).
f) Support the management entities for PBBN Auto Attach as specified in 12.34.

Each externally accessible AAB port that supports PBBN Auto Attach may:

g) Support the IEEE Std 802.1Q Organizationally Specific Management VID TLV (D.2.6)
h) Support SNMP MIB extensions for AAP (17.7.26).
i) Support SNMP MIB extensions for AAP LLDP TLVs (D.5.6).
j) Support YANG for AAP LLDP TLVs (D.6.6.7).

5.14 MAC Bridge conformance

Insert 5.14.2 after 5.14.1 as follows:

5.14.2 MAC Bridge requirements for PBBN Auto Attach (optional)

A MAC Bridge that conforms to the provisions for PBBN Auto Attach (Clause 50) shall:

a) Support Link Layer Discovery Protocol (LLDP) transmit and receive mode (IEEE Std 802.1AB).
b) Support the PBBN Auto Attach System TLV (D.2.17).
c) Support the PBBN Auto Attach Assignment TLV (D.2.18).
d) Support the AAD state machine (50.6).
e) Support the management entities for PBBN Auto Attach as specified in 12.34.

A MAC Bridge that conforms to the provisions for PBBN Auto Attach may:

f) Support SNMP MIB extensions for AAP (17.7.26).
g) Support SNMP MIB extensions for AAP LLDP TLVs (D.5.6).
h) Support YANG for AAP LLDP TLVs (D.6.6.7).

Insert 5.33 after 5.32 as follows:

5.33 End station requirements for PBBN Auto Attach (optional)

An end station that conforms to the provisions for PBBN Auto Attach (Clause 50) shall:

a) Support Link Layer Discovery Protocol (LLDP) transmit and receive mode (IEEE Std 802.1AB).
b) Support the PBBN Auto Attach System TLV (D.2.17).
c) Support the PBBN Auto Attach Assignment TLV (D.2.18).
d) Support the AAD state machine (50.6).
e) Support the management entities for PBBN Auto Attach as specified in 12.34.

An end station implementation that conforms to the provisions for PBBN Auto Attach may:

f) Support SNMP MIB extensions for AAP (17.7.26).
g) Support SNMP MIB extensions for AAP LLDP TLVs (D.5.6).
h) Support YANG for AAP LLDP TLVs (D.6.6.7).

12. Bridge Management

Insert 12.34 after 12.33, as follows:

12.34 Managed objects for PBBN Auto Attach

The conformance requirements for PBBN Auto Attach support are specified in 5.9.2, 5.10.3, 5.12.2, 5.14.2, and 5.33. PBBN Auto Attach is specified in Clause 50.

The PBBN Auto Attach specific managed objects defined here:

a) Provide managed objects for identifying and configuring an Auto Attach System (12.34.1).
b) Provide managed objects for configuring Bridge Ports, LAGs, and DRNIs for Auto Attach (12.34.2).
c) Provide managed objects to catalog discovered remote ports supporting Auto Attach (12.34.3).
d) Provide managed objects that indicate the desired VID to I-SID assignments for AADs and the operating assignments for AABs for each Bridge Port, LAG, or DRNI (12.34.4).
e) Provide per port PBBN Auto Attach statistics (12.34.5).

12.34.1 PBBN Auto Attach System objects

The PBBN Auto Attach System objects (Table 12-46) identify the AAS within the administrative domain, indicate the type of AAS, either AAB or one of the AAD types, and enable Auto Attach within the AAS. The aaSystemMAC uniquely identifies the system including all associated components.

Table 12-46—PBBN Auto Attach System objects

Name	Data type	Operations supported [a]	Conformance [b]	References
aaSystemEnable	Boolean	RW	B, D	50.4.2.1
aaSystemMAC	macAddress	R	B, D	50.4.2.4
aaSystemType	enumerated {AAB, CVLAN-AWARE-AAD, VLAN-UNAWARE-AAD, SVLAN-AWARE-AAD}	R	B, D	50.4.2.5
aaSystemResetTime	unsigned (1...1200)	RW	B, D	50.4.2.6

[a] R = Read only access; RW = Read/Write access.
[b] B = required for AAB; D= required for AAD.

In the case of an AAB a single aaSystemMAC identifies all I-components and T-components as well as all embedded S-components and C-components used to support external C-VLAN and S-VLAN interfaces. In the case where a pair of BEBs form a DRNI, the single aaSystemMAC identifies all components of both BEBs coupled to form a DRNI.

For an AAD the single aaSystemMAC identifies all end station and Bridge components within the AAD.

The object aaSystemEnable defaults to FALSE allowing configuration of the system before Auto Attach operation. Changing aaSystemEnable from FALSE to TRUE starts Auto Attach on all configured and enabled ports. Setting aaSystemEnable to FALSE breaks all attachments and disables the Auto Attach feature on all ports of the AAS.

12.34.2 PBBN Auto Attach port table

Each AAS has a port table that lists the ports that can be advertised by this system using the AAP. Each entry in the PBBN Auto Attach port table (Table 12-47) can be created, deleted, and updated by the system administrator to configure ports for Auto Attach.

Table 12-47—PBBN Auto Attach port table

Name	Data type	Operations supported [a]	Conformance [b]	References
aaPortId	If-ref	R	B, D	50.4.3.13
aaPortNetId	String	R	B, D	50.4.3.9, 50.4.1.3 portNetId [c]
aaPortEnable	Boolean	RW	B, D	50.4.3.1

[a] R = Read only access; RW = Read/Write access.
[b] B = required for AAB; D= required for AAD.
[c] This is the object within the referenced structure.

Each local port is uniquely identified within the AAS by its aaPortId and uniquely identified over the administrative domain by aaPortNetId. The aaPortNetId is composed of the aaSystemMAC and the aaPortID. The management object aaPortNetId reflects the content of the state machine variable aaOperLocSysTlv.portNetId.

The object aaPortEnable controls the state machine variable aaAdminPortEnable that enables/disables Auto Attach operation on this port. Setting aaPortEnable = FALSE forces the AAP state machines for this port to restart, detaching all active associations and clearing the current statistics.

12.34.3 PBBN Discovered Auto Attach Systems table

The PBBN Discovered Auto Attach Systems table (Table 12-48) object provides information about active remote Auto Attach ports identified by the Auto Attach Protocol (AAP) through the Auto Attach LLDP database system objects. The Discovered Auto Attach Systems table is read only.

Table 12-48—PBBN Discovered Auto Attach Systems table

Name	Data type	Operations supported [a]	Conformance [b]	References
aaDiscLocPortId	If-ref	R	B, D	50.4.3.13
aaDiscRemPortNetId	String	R	B, D	50.4.3.11, 50.4.1.3, D.2.17.10 portNetId [c]
aaDiscRemSystemType	Enumeration {AAB, CVLAN-AWARE-AAD, VLAN-UNAWARE-AAD, SVLAN-AWARE-AAD}	R	B, D	50.4.3.11, 50.4.1.3, D.2.17.7 sysType[c]
aaDiscRemPortTagging	Enumeration {TAG-ALL, TAG-OR-UNTAG, UNTAG-ONLY}	R	B, D	50.4.3.11, 50.4.1.3, D.2.17.8 portTagging[c]
aaDiscRemPortAssocState	Enumeration {NOT_READY, READY_TO_ASSOC, READY_TO_ATTACH, ASSOC_FAILED_TYPES, ASSOC_FAILED_TAGS, ASSOC_FAILED_TOPO, ASSOC_FAILED_OTHER, ASSOC_ATTACHED, ASSOC_STANDBY, ASSOC_INVALID}	R	B, D	50.4.3.11, 50.4.1.3, D.2.17.5 state[c]
aaDiscLocPortAssocState	Enumeration {NOT_READY, READY_TO_ASSOC, READY_TO_ATTACH, ASSOC_FAILED_TYPES, ASSOC_FAILED_TAGS, ASSOC_FAILED_TOPO, ASSOC_FAILED_OTHER, ASSOC_ATTACHED, ASSOC_STANDBY, ASSOC_INVALID}	R	B, D	50.4.3.9, 50.4.1.3, D.2.17.5 state [c]

[a] Operations Supported: R = Read only access; RW = Read/Write access.

[b] B = required for AAB; D= required for AAD.

[c] This is the object within the referenced structure.

The aaDiscLocPortId provides a reference to the local port, LAG, or DRNI where the remote attach port was discovered. The aaDiscRemPortNetId uniquely identifies the remote port, LAG, or DRNI discovered at the local port.

12.34.4 PBBN Auto Attach assignment table

There is one PBBN Auto Attach assignment table for each port configured in the Auto Attach port table (12.34.2). The Auto Attach assignment table is used by management to set the VLAN ID/I-SID assignments desired by the AAD and to monitor the current Auto Attach assignment requests received at the AAB. Entries in the Auto Attach assignment table can be created, updated, and deleted for the AAD, but are read only for the AAB.

Each entry of the Auto Attach assignment table (Table 12-49) contains three variables specifying the VID, I-SID, and status of each assignment. In an AAD the desired assignments are stored and read from the per port state machine variable aaAdminVidIsidAsgns[] (50.4.3.2). One aaAdminVidIsidAsgns[] exists for each local port identified by the Auto Attach port table.

Table 12-49—PBBN Auto Attach assignment table

Name	Data type	Operations supported [a]	Conformance [b]	References
aaVidIsidAsgnsVid	unsigned[1...4094]	R for AAB, RW for AAD	B, D	50.4.3.2 vid [c]
aaVidIsidAsgnsIsid	unsigned [1, 256...16777214]	R for AAB, RW for AAD	B, D	50.4.3.2 isid [c]
aaVidIsidAsgnsStatus	Enumerated { PENDING, ACCEPTED, REJECTED, REJ-RES-UNAVAIL, REJ-INVALID-VID, REJ-VLAN-UNAVAIL, REJ-INVALID-I-SID, REJ-I-SID-UNAVAIL, REJ-APP-ISSUE, REJ-NOT-ALLOWED	R	B, D	AAD 50.4.3.10, AAB 50.4.3.8, 50.4.1.1 asgnsArray[] status [c]

[a] R = Read only access; RW = Read/Write access.
[b] B = required for AAB; D= required for AAD.
[c] This is the object within the referenced structure.

The status indicated in the Auto Attach assignment table for each VID to I-SID assignment is determined from the state machine variable aaOperRemAsgnsTlv.asgnsArray.status (50.4.3.10) in AADs or from the state machine variable aaOperLocAsgnsTlv.asgns.status (50.4.3.8) in AABs. The status at the AAD is the status from the VID/I-SID assignments reported in the remote LLDP assignment objects received from the AAB. The status at the AAB is the status determined by the AAB for the VID/I-SID assignment requests received from the AAD. At the AAD, if no remote LLDP assignment objects exist in the LLDP database or if the VID/I-SID assignment is not listed in the remote AAB LLDP database objects, then the status is set to PENDING in the Auto Attach assignment table.

The AAD administrator can create new entries, delete entries, and update entries in the Auto Attach assignment table. When the administrator deletes, adds, or updates an entry in the Auto Attach assignment table the AAP state machine variable aaAdminVidIsidAsgns[] associated with the affected port is updated to match the Auto Attach assignment table. When a new entry is created or updated, the new VID and I-SID is provided by management and the status begins as PENDING and then transitions to reflect the current status indicated in the remote AAB LLDP objects.

In an AAB the Auto Attach assignment table is read only and reflects the current assignments within the local LLDP assignment database. The Auto Attach assignment table contains the status, VID, and I-SID (D.2.18.6, D.2.18.7, D.2.18.8) from the local LLDP assignment database objects.

12.34.5 PBBN Auto Attach statistics table

There is one PBBN Auto Attach statistics table for each port configured in the Auto Attach port table (Table 12-47). Each Auto Attach statistics table (Table 12-50) provides per port statistics. All objects of the Auto Attach statistics table are read only. The contents of the table is maintained by the Auto Attach state machines (50.6, 50.7). All statistic counters on all ports are cleared on system reset.

Table 12-50—PBBN Auto Attach statistics table

Name	Data type	Operations supported [a]	Conformance [b]	References
aaStatsAssocAttached	Counter32	R	B, D	50.4.3.18
aaStatsAssocFailed	Counter32	R	B, D	50.4.3.19
aaStatsAssocReset	Counter32	R	B, D	50.4.3.20
aaStatsAssocStandby	Counter32	R	B, D	50.4.3.21
aaStatsVidIsidAsgnsAccepted	Counter32	R	B, D	50.4.3.22
aaStatsVidIsidAsgnsRejected	Counter32	R	B, D	50.4.3.23
aaStatsVidIsidAssgnsRequested	Counter32	R	B, D	50.4.3.24
aaStatsVidIsidAsgnsWithdrawn	Counter32	R	B, D	50.4.3.25

[a] R = Read only access; RW = Read/Write access.
[b] B = required for AAB; D= required for AAD.

The counters provided are as follows:

a) aaStatsAssocAttached: Counts the number of associations that were attached.
b) aaStatsAssocFailed: Counts the number of associations that failed.
c) aaStatsAssocReset: Counts the number of association resets.
d) aaStatsAssocStandby: Counts the number of associations that were standby.
e) aaStatsVidIsidAsgnsRequested: Counts the number of new VID to I-SID assignments requested.
f) aaStatsVidIsidAsgnsAccepted: Counts the number of VID to I-SID assignment requests accepted.
g) aaStatsVidIsidAsgnsRejected: Counts the number of VID to I-SID assignment requests rejected.
h) aaStatsVidIsidAsgnsWithdrawn: Counts the number of VID to I-SID assignment requests withdrawn.

17. Management Information Base (MIB)

17.2 Structure of the MIB

Insert a new row at the end of Table 17-1 as follows (unchanged rows not shown):

Table 17-1—IEEE 802.1Q MIB modules

Module	Structure, relationships, security considerations, module definition	Managed functionality	Initial functionality and MIB specification
IEEE8021-PBBN-AA-MIB	17.2.26, 17.3.26, 17.4.26, 17.7.26	Clause 50	IEEE Std 802.1Qcj

Insert 17.2.26 after 17.2.25 as follows:

17.2.26 Structure of the IEEE8021-PBBN-AA-MIB

The IEEE8021-PBBN-AA-MIB module provides objects to configure and manage the Auto Attach Protocol (AAP, Clause 50).

Table 17-32—IEEE8021-PBBN-AA-MIB Structure

IEEE8021-PBBN-AA-MIB table/objects	Reference
ieee8021AaConfig	
ieee8021AaSystemEnable	aaSystemEnable, 12.34.1, 50.4.2.1
ieee8021AaSystemType	aaSystemType, 12.34.1, 50.4.2.5
ieee8021AaSystemMAC	aaSystemMAC, 12.34.1, 50.4.2.4
ieee8021AaSystemResetTime	aaSystemResetTime, 12.34.1, 50.4.2.6
ieee8021AaPortTable	PBBN Auto Attach port table, 12.34.2
ieee8021AaPortEntry	12.34.2
ieee8021AaPortIfIndex*	aaPortId, 12.34.2, 50.4.3.13
ieee8021AaPortNetId*	aaPortNetId, 12.34.2, 50.4.3.9, 50.4.1.3
ieee8021AaPortEnable	aaPortEnable, 12.34.2, 50.4.3.1
ieee8021AaPortRowStatus	—
ieee8021AaDiscSystemsTable	PBBN Discovered Auto Attach Systems table, 12.34.3
ieee8021AaDiscSystemsEntry	12.34.3
ieee8021AaDiscLocPortIfIndex*	aaDiscLocPortId, 12.34.2, 50.4.3.13
ieee8021AaDiscRemSystemType	aaDiscRemSystemType, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscRemPortNetId*	aaDiscRemPortNetId, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscRemPortTagging	aaDiscRemPortTagging, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscRemPortAssocState	aaDiscRemPortAssocState, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscLocPortAssocState	aaDiscLocPortAssocState, 12.34.3, 50.4.3.9, 50.4.1.3

Table 17-32—IEEE8021-PBBN-AA-MIB Structure *(continued)*

IEEE8021-PBBN-AA-MIB table/objects	Reference
ieee8021AaIsidVidAsgnsTable	PBBN Auto Attach assignment table, 12.34.4
ieee8021AaIsidVidAsgnsEntry	12.34.4
ieee8021AaIsidVidAsgnsIfIndex*	aaPortId, 12.34.2, 50.4.3.13
ieee8021AaIsidVidAsgnsIsid*	aaVidIsidAsgnsIsid, 12.34.4, 50.4.3.2, 50.4.1.4
ieee8021AaIsidVidAsgnsVlan*	aaVidIsidAsgnsVid, 12.34.4, 50.4.3.2, 50.4.1.4
ieee8021AaIsidVidAsgnsStatus	aaVidIsidAsgnsStatus, 12.34.4, 50.4.3.10, 50.4.3.8
ieee8021AaIsidVidAsgnsRowStatus	—
ieee8021AaStats	
ieee8021AaStatsTable	PBBN Auto Attach statistics table, 12.34.5
ieee8021AaStatsEntry	12.34.5
ieee8021AaStatsPortIfIndex*	aaPortId, 12.34.2, 50.4.3.13
ieee8021AaStatsAssocAttached	aaStatsAssocAttached, 12.34.5, 50.4.3.18
ieee8021AaStatsAssocFailed	aaStatsAssocFailed, 12.34.5, 50.4.3.19
ieee8021AaStatsAssocReset	aaStatsAssocReset, 12.34.5, 50.4.3.20
ieee8021AaStatsAssocStandby	aaStatsAssocStandby, 12.34.5, 50.4.3.21
ieee8021AaStatsAsgnsRequested	aaStatsVidIsidAsgnsRequested, 12.34.5, 50.4.3.24
ieee8021AaStatsAsgnsAccepted	aaStatsVidIsidAsgnsAccepted, 12.34.5, 50.4.3.22
ieee8021AaStatsAsgnsRejected	aaStatsVidIsidAsgnsRejected, 12.34.5, 50.4.3.23
ieee8021AaStatsAsgnsWithdrawn	aaStatsVidIsidAsgnsWithdrawn, 12.34.5, 50.4.3.25
ieee8021AaNotifyObjects	
ieee8021AaNotifications	—
ieee8021AaDiscoveredSystem	—
ieee8021AaDiscRemSystemType	aaDiscRemSystemType, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscRemPortNetId	aaDiscRemPortNetId, 12.34.3, 50.4.3.11, 50.4.1.3
ieee8021AaDiscSystemsDescr	IEEE Std 802.1AB-2016, 8.5.7.2
ieee8021AaDiscSystemsMgmtOid	IEEE Std 802.1AB-2016, 8.5.9.8
ieee8021AaNotifyObjects	—
ieee8021AaDiscSystemsDescr	IEEE Std 802.1AB-2016, 8.5.7.2
ieee8021AaDiscSystemsMgmtOid	IEEE Std 802.1AB-2016, 8.5.9.8

*This object is an INDEX of the table in which it resides.

17.3 MIB module relationships

Insert 17.3.26 after 17.3.25, as follows:

17.3.26 Relationship of the IEEE8021-PBBN-AA-MIB to other MIB modules

The IEEE8021-PBBN-AA-MIB provides objects that extend the functionality of IEEE802.1AB LLDP.

17.4 Security considerations

Insert 17.4.26 after 17.4.25, as follows:

17.4.26 Security considerations of the IEEE8021-PBBN-AA-MIB

There are a number of management objects defined in the IEEE8021-PBBN-AA-MIB module that have a MAX-ACCESS clause of read-write. Such objects can 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.

The following tables and objects in the IEEE8021-PBBN-AA-MIB can be misconfigured to interfere with the operation of the forwarding and queuing mechanisms in a manner that would be detrimental to the transmission of scheduled traffic:

ieee8021AaSystemEnable
ieee8021AaPortRowStatus
ieee8021AaPortEnable
ieee8021AaIsidVidAsgnsRowStatus
ieee8021AaIsidVidAsgnsIsid
ieee8021AaIsidVidAsgnsVlan

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not accessible) can be considered sensitive or vulnerable in some network environments. It is thus important to control all types of access (including GET and/or NOTIFY) to these objects and possibly to encrypt the values of these objects when sending them over the network via SNMP.

17.7 MIB modules[8] [9]

Insert 17.7.26 after 17.7.25 as follows:

17.7.26 Definitions for the IEEE8021-PBBN-AA-MIB module

```
IEEE8021-PBBN-AA-MIB DEFINITIONS ::= BEGIN

-- ============================================================
-- MIB for support of PBBN Auto Attach to Provider Backbone
-- Bridging (PBB) services.
-- ============================================================

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE, Integer32, Gauge32, Unsigned32, Counter32,
    NOTIFICATION-TYPE
        FROM SNMPv2-SMI
    SnmpAdminString
            FROM SNMP-FRAMEWORK-MIB
    RowStatus, TruthValue, MacAddress
            FROM SNMPv2-TC
    MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
            FROM SNMPv2-CONF;

ieee8021PbbnAutoAttachMib MODULE-IDENTITY
    LAST-UPDATED "202309210000Z" -- September 21, 2023
    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-chairs@ieee.org"
    DESCRIPTION
        "The PBBN Auto Attach MIB module for configuration and statistics support.

        Unless otherwise indicated, the references in this MIB module are to
        IEEE Std 802.1Q Bridges and Bridged Networks:
        IEEE Std 802.1Q-2022 as amended by IEEE Std 802.1Qcz, IEEE Std 802.1Qcw,
        and IEEE Std 802.1Qcj.

        Copyright (C) IEEE (2023).
        This version of this MIB module is part of IEEE Std 802.1Q;
        see that standard for full legal notices."

    REVISION    "202309210000Z" -- September 21, 2023
    DESCRIPTION
        "Initial version published as part of IEEE Std 802.1Qcj."

    ::= { iso (1) iso-identified-organization (3) ieee (111)
            standards-association-numbered-series-standard1s (2)
            lan-man-stds (802) ieee802dot1(1)
            ieee802dot1mibs(1)  ieee8021PbbnAutoAttachMib(37) }

ieee8021AaNotifications OBJECT IDENTIFIER ::= { ieee8021PbbnAutoAttachMib 0 }
ieee8021AaObjects       OBJECT IDENTIFIER ::= { ieee8021PbbnAutoAttachMib 1 }
ieee8021AaNotifyObjects OBJECT IDENTIFIER ::= { ieee8021PbbnAutoAttachMib 2 }

ieee8021AaConfig    OBJECT IDENTIFIER ::= { ieee8021AaObjects 1 }
ieee8021AaStats     OBJECT IDENTIFIER ::= { ieee8021AaObjects 2 }
```

[8] *Copyright release for MIBs:* Users of this standard may freely reproduce the MIB modules in this standard so that they can be used for their intended purpose.

[9] An ASCII version of this MIB module is attached to the PDF version of this standard, and can be obtained by Web browser from the IEEE 802.1 Website at https://1.ieee802.org/mib-modules/.

```
-- ================================================================
-- Auto Attach scalar attributes
-- ================================================================

ieee8021AaSystemEnable OBJECT-TYPE
    SYNTAX         INTEGER {
                       enabled(1),
                       disabled(2)
                   }
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "Exports the status of the Auto Attach service on this system."
    ::= { ieee8021AaConfig 1 }

ieee8021AaSystemType OBJECT-TYPE
    SYNTAX         INTEGER {
                       aaBeb(1),
                       aaDeviceVlanAware(2),
                       aaDeviceVlanUnaware(3)
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Defines the AAS type indicating the services supported by the system.
        May be read-only on platforms that only support a single system type."
    ::= { ieee8021AaConfig 2 }

ieee8021AaSystemMAC OBJECT-TYPE
    SYNTAX         MacAddress
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Defines the Auto Attach System MAC Address used to uniquely identify
        this system."
    ::= { ieee8021AaConfig 3 }

ieee8021AaSystemResetTime OBJECT-TYPE
    SYNTAX         Integer32 (1..1200)
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "AAP reset time delay used to re-synchronize the associated AASs"
    ::= { ieee8021AaConfig 4 }

-- ================================================================
-- Auto Attach VID/I-SID assignment table support
-- ================================================================

ieee8021AaVidIsidAsgnsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Ieee8021AaVidIsidAsgnsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains the VID <-> I-SID assignments for the
        Auto Attach application."
    ::= { ieee8021AaConfig 5 }

ieee8021AaVidIsidAsgnsEntry OBJECT-TYPE
    SYNTAX      Ieee8021AaVidIsidAsgnsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Information about the I-SID <-> VID assignments for the
        Auto Attach application."
    INDEX { ieee8021AaVidIsidAsgnsIfIndex,
            ieee8021AaVidIsidAsgnsIsid,
            ieee8021AaVidIsidAsgnsVid }
    ::= { ieee8021AaVidIsidAsgnsTable 1 }
```

```
Ieee8021AaVidIsidAsgnsEntry ::= SEQUENCE {
    ieee8021AaVidIsidAsgnsIfIndex    Integer32,
    ieee8021AaVidIsidAsgnsIsid       Integer32,
    ieee8021AaVidIsidAsgnsVid       Integer32,
    ieee8021AaVidIsidAsgnsStatus     INTEGER,
    ieee8021AaVidIsidAsgnsRowStatus  RowStatus }

ieee8021AaVidIsidAsgnsIfIndex OBJECT-TYPE
    SYNTAX         Integer32 (0..65535)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "Interface identifier component of the I-SID <-> VID assignment."
    ::= { ieee8021AaVidIsidAsgnsEntry 1 }

ieee8021AaVidIsidAsgnsIsid OBJECT-TYPE
    SYNTAX         Integer32 (1256..16777214)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "I-SID component of the I-SID <-> VID assignment."
    ::= { ieee8021AaVidIsidAsgnsEntry 2 }

ieee8021AaVidIsidAsgnsVid OBJECT-TYPE
    SYNTAX         Integer32 (1..4094)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "VLAN ID component of the I-SID <-> VID assignment."
    ::= { ieee8021AaVidIsidAsgnsEntry 3 }

ieee8021AaVidIsidAsgnsStatus OBJECT-TYPE
    SYNTAX         INTEGER {
                       pending(1),
                       accepted(2),
                       rejected(3),
                       rejectedAutoAttachResourcesUnavailable(4),
                       rejectedInvalidVLANID(5),
                       rejectedVLANResourcesUnavailable(6),
                       rejectedInvalidISID(7),
                       rejectedISIDResourcesUnavailable(8),
                       rejectedApplicationInteractionIssue(9),
                       rejectedAssignmentNotAllowed(10)
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "The current state of the Auto Attach I-SID <-> VID assignment.
        Rejections can be from multiple causes including resources
        unavailable, duplicate, application interaction, etc."
    ::= { ieee8021AaVidIsidAsgnsEntry 4 }

ieee8021AaVidIsidAsgnsRowStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-write
    STATUS         current
    DESCRIPTION
        "This object allows entries to be created and deleted in the
        ieee8021AaVidIsidAsgnsTable."
    ::= { ieee8021AaVidIsidAsgnsEntry 5 }
```

```
-- ================================================================
-- Auto Attach port table support
-- ================================================================

ieee8021AaPortTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Ieee8021AaPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains port-specific data for the
        Auto Attach application."
    ::= { ieee8021AaConfig 6 }

ieee8021AaPortEntry OBJECT-TYPE
    SYNTAX      Ieee8021AaPortEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Port-specific data related to the Auto Attach application."
    INDEX { ieee8021AaPortIfIndex,
            ieee8021AaPortNetId }
    ::= { ieee8021AaPortTable 1 }

Ieee8021AaPortEntry ::= SEQUENCE {
    ieee8021AaPortIfIndex        Integer32,
    ieee8021AaPortNetId          OCTET STRING,
    ieee8021AaPortEnable         INTEGER,
    ieee8021AaPortRowStatus      RowStatus }

ieee8021AaPortIfIndex OBJECT-TYPE
    SYNTAX         Integer32 (0..65535)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "IfIndex on the interface (port) identifier attribute."
    ::= { ieee8021AaPortEntry 1 }

ieee8021AaPortNetId OBJECT-TYPE
    SYNTAX         OCTET STRING (SIZE (0..12))
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Identifies an AAS port uniquely within the administrative domain.
        The object is the concatenation of the ieee8021AaSystemMAC and
        ieee8021AaPortIfIndex. The identified port can be simple, a LAG,
        or a DRNI."
    ::= { ieee8021AaPortEntry 2 }

ieee8021AaPortEnable OBJECT-TYPE
    SYNTAX         INTEGER {
                       enabled(1),
                       disabled(2)
                   }
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
        "The current port Auto Attach Protocol enable. Indicates whether
        Auto Attach TLVs will be included in LLDPDUs generated on
        the port (enabled) or not (disabled)."
    ::= { ieee8021AaPortEntry 3 }

ieee8021AaPortRowStatus OBJECT-TYPE
    SYNTAX         RowStatus
    MAX-ACCESS     read-create
    STATUS         current
    DESCRIPTION
        "This object allows entries to be created, deleted, and modified in the
        ieee8021AaPortTable, if the operation is supported by the agent."
    ::= { ieee8021AaPortEntry 4 }
```

```
-- ================================================================
-- Discovered Auto Attach Systems table support
-- ================================================================

ieee8021AaDiscSystemsTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF Ieee8021AaDiscSystemsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "This table contains details about locally connected AASs that have
        been discovered."
    ::= { ieee8021AaConfig 7 }

ieee8021AaDiscSystemsEntry OBJECT-TYPE
    SYNTAX      Ieee8021AaDiscSystemsEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
        "Contains details about a locally connected AAS."
    INDEX { ieee8021AaDiscLocPortIfIndex,
            ieee8021AaDiscRemPortNetId }
    ::= { ieee8021AaDiscSystemsTable 1 }

Ieee8021AaDiscSystemsEntry ::= SEQUENCE {
    ieee8021AaDiscLocPortIfIndex       Integer32,
    ieee8021AaDiscRemSystemType        INTEGER,
    ieee8021AaDiscRemPortNetId         OCTET STRING,
    ieee8021AaDiscRemPortTagging       BITS,
    ieee8021AaDiscRemPortAssocState    INTEGER,
    ieee8021AaDiscLocPortAssocState    INTEGER }

ieee8021AaDiscLocPortIfIndex OBJECT-TYPE
    SYNTAX         Integer32 (0..65535)
    MAX-ACCESS     not-accessible
    STATUS         current
    DESCRIPTION
        "Identifies the interface through which the AAS
    represented by this entry was discovered."
    ::= { ieee8021AaDiscSystemsEntry 1 }

ieee8021AaDiscRemSystemType OBJECT-TYPE
    SYNTAX         INTEGER {
                       aaBeb(1),
                       aaDeviceCVlanAware(2),
                       aaDeviceVlanUnaware(3),
                       aaDeviceSVlanAware(4)
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Identifies the AAS type as advertised through LLDP."
    ::= { ieee8021AaDiscSystemsEntry 2 }

ieee8021AaDiscRemPortNetId OBJECT-TYPE
    SYNTAX         OCTET STRING (SIZE (0..12))
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Identifies an AAS port uniquely within the administrative domain.
        The object is the concatenation of the ieee8021AaSystemMAC and
        port IfIndex. The identified port can be simple, a LAG, or a DRNI."
    ::= { ieee8021AaDiscSystemsEntry 3 }
```

```
ieee8021AaDiscRemPortTagging OBJECT-TYPE
    SYNTAX         BITS {
                       trafficTagged(0),
                       trafficTaggedAndUntagged(1),
                       trafficUntaggedOnly(2)
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "Exports the tagging data associated with the discovered
        Auto Attach System as advertised through LLDP. When bit 0 is set
        to 0, all traffic is tagged on the link; when bit 0 is set to 1,
        the link supports both tagged and untagged traffic. When bit 2 is
        set to 1, the link supports only untagged traffic."
    ::= { ieee8021AaDiscSystemsEntry 4 }

ieee8021AaDiscRemPortAssocState OBJECT-TYPE
    SYNTAX         INTEGER {
                notReady(0), -- initializing, not ready
                readyToAssoc(1), -- ready to associate
                readyToAttach(2), -- viable partner
                assocAttached(3), -- port is active
                assocFailTypes(18), -- System type mismatched
                assocStandby(19), -- at AAD, active elsewhere
                assocFailTags(34), -- Tagging mismatched
                assocInvalid(35), -- at AAB, AAD active elsewhere
                assocFailTopo(50), -- Multipoint link detected
                assocFailOther(66) -- Parsing or resourcing error
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "The current remote operational status for the associated interface."
    ::= { ieee8021AaDiscSystemsEntry 5 }

ieee8021AaDiscLocPortAssocState OBJECT-TYPE
    SYNTAX      INTEGER {
                notReady(0), -- initializing, not ready
                readyToAssoc(1), -- advertising ready to associate
                readyToAttach(2), -- viable partner
                assocAttached(3), -- port is active
                assocFailTypes(18), -- System type mismatched
                assocStandby(19), -- at AAD, active elsewhere
                assocFailTags(34), -- Tagging mismatched
                assocInvalid(35), -- at AAB, AAD active elsewhere
                assocFailTopo(50), -- Multipoint link detected
                assocFailOther(66) -- Parsing or resourcing error
                   }
    MAX-ACCESS     read-only
    STATUS         current
    DESCRIPTION
        "The current local operational status for the associated interface."
    ::= { ieee8021AaDiscSystemsEntry 6 }

-- ================================================================
-- AAS statistics and error counters table
-- ================================================================

ieee8021AaStatsTable  OBJECT-TYPE
    SYNTAX        SEQUENCE OF Ieee8021AaStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "This table contains Auto Attach port statistics data."
    ::= { ieee8021AaStats 1 }
```

```
ieee8021AaStatsEntry OBJECT-TYPE
    SYNTAX        Ieee8021AaStatsEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "Each entry contains Auto Attach operational statistics for a
    specific Auto Attach System port. Ports are identified by their MIB II
    ifIndex value."
    INDEX         { ieee8021AaStatsPortIfIndex }
    ::= { ieee8021AaStatsTable 1 }

Ieee8021AaStatsEntry ::= SEQUENCE {
    ieee8021AaStatsPortIfIndex            Integer32,
    ieee8021AaStatsAssocAttached          Counter32,
    ieee8021AaStatsAsgnsRequested         Counter32,
    ieee8021AaStatsAsgnsAccepted          Counter32,
    ieee8021AaStatsAsgnRejected           Counter32,
    ieee8021AaStatsAssocFailed            Counter32,
    ieee8021AaStatsAsgnsWithdrawn         Counter32,
    ieee8021AaStatsAssocReset             Counter32,
    ieee8021AaStatsAssocStandby           Counter32 }

ieee8021AaStatsPortIfIndex OBJECT-TYPE
    SYNTAX        Integer32 (0..65535)
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "An index value that uniquely identifies a port. This value
    corresponds to a MIB II ifIndex."
    ::= { ieee8021AaStatsEntry 1 }

ieee8021AaStatsAssocAttached OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of PBBN Auto Attach System TLVs received on the
        identified port."
    ::= { ieee8021AaStatsEntry 2 }

ieee8021AaStatsAsgnsRequested OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of I-SID/VID bindings received in
        PBBN Auto Attach Assignment TLVs on the identified port."
    ::= { ieee8021AaStatsEntry 3 }

ieee8021AaStatsAsgnsAccepted OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of I-SID/VID bindings from received
        PBBN Auto Attach Assignment TLVs that are accepted (activated)
        on the identified port. Counter is incremented when the binding
        transitions from a non-accepted state (e.g., 'pending', 'rejected')
        to the accepted state. The AAD counts the number received from the
        AAB while the AAB counts the number transmitted to the AAD."
    ::= { ieee8021AaStatsEntry 4 }
```

```
ieee8021AaStatsAsgnRejected OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of I-SID/VID bindings from received Auto Attach
        Assignment TLVs that are rejected on the identified port. Counter is
        incremented when the binding transitions from a non-rejected state
        (e.g., 'pending', 'accepted') to the rejected state. The AAD counts
        the number received from the AAB while the AAB counts the number
        transmitted to the AAD."
    ::= { ieee8021AaStatsEntry 5 }

ieee8021AaStatsAssocFailed OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of discovered AAS from received AA System TLVs
        that have failed association or attachment due to errors such as
        system mismatches, multi-point link, incompatible tagging, etc."
    ::= { ieee8021AaStatsEntry 6 }

ieee8021AaStatsAsgnsWithdrawn OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of I-SID/VID bindings from received PBBN
        Auto Attach Assignment TLVs that have been deleted on the identified
        port. Counter is only incremented when bindings are deleted for
        reasons other than expiration."
    ::= { ieee8021AaStatsEntry 7 }

ieee8021AaStatsAssocReset OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of discovered AAS from received AA System TLVs
        that have reset an association or attachment due deleted
        discovered system TLVs, etc. This condition is most commonly a result
        of an AAS being reset or powered down."
    ::= { ieee8021AaStatsEntry 8 }

ieee8021AaStatsAssocStandby OBJECT-TYPE
    SYNTAX        Counter32
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "Indicates the number of discovered AAS from received AA System TLVs
        that have had an association in standby."
    ::= { ieee8021AaStatsEntry 9 }

-- ================================================================
-- Auto Attach notification objects
-- ================================================================

ieee8021AaDiscSystemsDescr OBJECT-TYPE
    SYNTAX         SnmpAdminString (SIZE(0..255))
    MAX-ACCESS     accessible-for-notify
    STATUS         current
    DESCRIPTION
        "The string value used to identify the LLDP system description of the
        remote system distributed through LLDPDUs."
    REFERENCE
        "IEEE 802.1AB-2016 8.5.7.2"
    ::= { ieee8021AaNotifyObjects 1 }

ieee8021AaDiscSystemsMgmtOid OBJECT-TYPE
    SYNTAX         OBJECT IDENTIFIER
    MAX-ACCESS     accessible-for-notify
```

```
    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 distributed through LLDP."
    REFERENCE
        "IEEE 802.1AB-2016 8.5.9.8"
    ::= { ieee8021AaNotifyObjects 2 }

-- ================================================================
-- Auto Attach notifications
-- ================================================================

ieee8021AaDiscoveredSystem NOTIFICATION-TYPE
    OBJECTS        {
                       ieee8021AaDiscRemSystemType,
                       ieee8021AaDiscRemPortNetId,
                       ieee8021AaDiscSystemsDescr,
                       ieee8021AaDiscSystemsMgmtOid
                   }
    STATUS         current
    DESCRIPTION
        "This notification is generated when an AAS is discovered."
    ::= { ieee8021AaNotifications  1 }


-- ================================================================================
-- IEEE 802.1 - Auto Attach conformance information
-- ================================================================================

ieee8021AaConformance OBJECT IDENTIFIER ::= { ieee8021PbbnAutoAttachMib 4 }

ieee8021AaCompliances
    OBJECT IDENTIFIER ::= { ieee8021AaConformance 1 }
ieee8021AaGroups
    OBJECT IDENTIFIER ::= { ieee8021AaConformance 2 }


--
-- Auto Attach - compliance statements
--

ieee8021AaCompliance MODULE-COMPLIANCE
    STATUS        current
    DESCRIPTION
         "A compliance statement for SNMP entities that implement
         the IEEE 802.1Qcj Auto Attach MIB.

         This group is mandatory for agents that implement the
         Auto Attach aaSet TLV set."
    MODULE        -- this module
        MANDATORY-GROUPS  { ieee8021AaGroup }
    ::= { ieee8021AaCompliances 1 }


--
-- Auto Attach - MIB groupings
--

ieee8021AaGroup  OBJECT-GROUP
    OBJECTS {
        ieee8021AaSystemEnable,
        ieee8021AaSystemType,
        ieee8021AaSystemMAC,
        ieee8021AaVidIsidAsgnsStatus,
        ieee8021AaVidIsidAsgnsRowStatus,
        ieee8021AaPortNetId,
        ieee8021AaPortEnable,
        ieee8021AaDiscRemPortNetId,
        ieee8021AaStatsAssocAttached,
        ieee8021AaStatsAsgnsRequested,
        ieee8021AaStatsAsgnsAccepted,
```

```
        ieee8021AaStatsAsgnRejected,
        ieee8021AaDiscSystemsDescr,
        ieee8021AaDiscSystemsMgmtOid
      }
    STATUS  current
    DESCRIPTION
        "The collection of objects that support configuration of the PBBN
        Auto Attach service"
    ::= { ieee8021AaGroups 1 }

END
```

48. YANG Data Models

48.6 YANG modules[10] [11] [12]

48.6.2 The ieee802-dot1q-types YANG module

Delete the YANG module in 48.6.2.

Insert the following YANG module in 48.6.2:

```
module ieee802-dot1q-types {
  yang-version "1.1";
  namespace urn:ieee:std:802.1Q:yang:ieee802-dot1q-types;
  prefix dot1q-types;
  import ietf-yang-types {
    prefix yang;
  }
  organization
    "IEEE 802.1 Working Group";
  contact
    "WG-URL: http://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-chairs@ieee.org";
  description
    "Common types used within dot1Q-bridge modules.

    Copyright (C) IEEE (2023).

    This version of this YANG module is part of IEEE Std 802.1Q; see the
    standard itself for full legal notices.";
  revision 2023-10-26 {
    description
      "Published as part of IEEE Std 802.1Qcj-2023.  This revision adds
      the isid-type.

      The following reference statement identifies each referenced IEEE
      Standard as updated by applicable amendments.";
    reference
      "IEEE Std 802.1Q Bridges and Bridged Networks:
      IEEE Std 802.1Q-2022, IEEE Std 802.1Qcz-2023, IEEE Std 802.1Qcw-2023,
      IEEE Std 802.1Qcj-2023.
      IEEE Std 802 Overview and Architecture: IEEE Std 802-2014.";
  }
  revision 2023-10-22 {
    description
      "Published as part of IEEE Std 802.1Qcw-2023.

      The following reference statement identifies each referenced IEEE
      Standard as updated by applicable amendments.";
    reference
      "IEEE Std 802.1Q Bridges and Bridged Networks:
```

[10] Copyright release for YANG: Users of this standard may freely reproduce the YANG modules contained in this standard so that they can be used for their intended purpose.

[11] An ASCII version of each YANG module is attached to the PDF of this standard and can also be obtained from the IEEE 802.1 Website at https://1.ieee802.org/yang-modules/.

[12] References in this standard's YANG module definitions are not clickable, as each module has been incorporated unchanged after development and verification using YANG tools.

```
    IEEE Std 802.1Q-2022, IEEE Std 802.1Qcz-2023, IEEE Std 802.1Qcw-2023.
    IEEE Std 802 Overview and Architecture:
    IEEE Std 802-2014.";
}
revision 2023-07-03 {
  description
    "Published as part of IEEE Std 802.1Qcz-2023.";
  reference
    "IEEE Std 802.1Qcz-2023, Bridges and Bridged Networks - Congestion
    Isolation.";
}
revision 2022-10-29 {
  description
    "Published as part of IEEE Std 802.1Q-2022.";
  reference
    "IEEE Std 802.1Q-2022, Bridges and Bridged Networks.";
}
revision 2020-06-04 {
  description
    "Published as part of IEEE Std 802.1Qcx-2020.";
  reference
    "IEEE Std 802.1Qcx-2020, Bridges and Bridged Networks - YANG Data
    Model for Connectivity Fault Management.";
}
revision 2018-03-07 {
  description
    "Published as part of IEEE Std 802.1Q-2018.";
  reference
    "IEEE Std 802.1Q-2018, Bridges and Bridged Networks.";
}
identity dot1q-vlan-type {
  description
    "Base identity from which all 802.1Q VLAN tag types are derived.";
}
identity c-vlan {
  base dot1q-vlan-type;
  description
    "An 802.1Q Customer VLAN, using the 81-00 EtherType.";
  reference
    "5.5 of IEEE Std 802.1Q";
}
identity s-vlan {
  base dot1q-vlan-type;
  description
    "An 802.1Q Service VLAN, using the 88-A8 EtherType originally
    introduced in 802.1ad, and incorporated into 802.1Q (2011)";
  reference
    "5.6 of IEEE Std 802.1Q";
}
identity transmission-selection-algorithm {
  description
    "Specify the transmission selection algorithms of IEEE Std 802.1Q
    Table 8-6";
}
identity strict-priority {
  base transmission-selection-algorithm;
  description
    "Indicates the strict priority transmission selection algorithm.";
  reference
    "Table 8-6 of IEEE Std 802.1Q";
}
identity credit-based-shaper {
  base transmission-selection-algorithm;
  description
    "Indicates the credit-based shaper transmission selection algorithm.";
  reference
    "Table 8-6 of IEEE Std 802.1Q";
}
identity enhanced-transmission-selection {
  base transmission-selection-algorithm;
  description
    "Indicates the enhanced transmission selection algorithm.";
```

```
  reference
    "Table 8-6 of IEEE Std 802.1Q";
}
identity asynchronous-traffic-shaping {
  base transmission-selection-algorithm;
  description
    "Indicates the asynchronous transmission selection algorithm.";
  reference
    "Table 8-6 of IEEE Std 802.1Q";
}
identity vendor-specific {
  base transmission-selection-algorithm;
  description
    "Indicates a vendor specific transmission selection algorithm.";
  reference
    "Table 8-6 of IEEE Std 802.1Q";
}
identity type-of-operation {
  description
    "Represents the operation type (name).";
}
typedef name-type {
  type string {
    length "0..32";
  }
  description
    "A text string of up to 32 characters, of locally determined
    significance.";
}
typedef port-number-type {
  type uint32 {
    range "1..4095";
  }
  description
    "The port number of the Bridge port for which this entry contains
    Bridge management information.";
}
typedef priority-type {
  type uint8 {
    range "0..7";
  }
  description
    "A range of priorities from 0 to 7 (inclusive). The Priority Code
    Point (PCP) is a 3-bit field that refers to the class of service
    associated with an 802.1Q VLAN tagged frame. The field specifies a
    priority value between 0 and 7, these values can be used by quality of
    service (QoS) to prioritize different classes of traffic.";
}
typedef num-traffic-class-type {
  type uint8 {
    range "1..8";
  }
  description
    "The number of traffic classes supported or participating in a
    particular feature. There are between 1 and 8 supported traffic
    classes defined by IEEE Std 802.1Q.";
}
typedef vid-range-type {
  type string {
    pattern
      "([1-9]"+
      "[0-9]{0,3}"+
      "(-[1-9][0-9]{0,3})?"+
      "(,[1-9][0-9]{0,3}(-[1-9][0-9]{0,3})?)*)";
  }
  description
    "A list of VLAN IDs, or non overlapping VLAN ranges, in ascending
    order, between 1 and 4094.

    This type is used to match an ordered list of VLAN IDs, or contiguous
    ranges of VLAN IDs. Valid VLAN IDs must be in the range 1 to 4094, and
    included in the list in non overlapping ascending order.
```

```
    For example: 1,10-100,250,500-1000";
}
typedef vlanid {
  type uint16 {
    range "1..4094";
  }
  description
    "The vlanid type uniquely identifies a VLAN. This is the 12-bit
    VLAN-ID used in the VLAN Tag header. The range is defined by the
    referenced specification. This type is in the value set and its
    semantics equivalent to the VlanId textual convention of the SMIv2.";
}
typedef vlan-index-type {
  type uint32 {
    range "1..4094 | 4096..4294967295";
  }
  description
    "A value used to index per-VLAN tables. Values of 0 and 4095 are not
    permitted. The range of valid VLAN indices. If the value is greater
    than 4095, then it represents a VLAN with scope local to the
    particular agent, i.e., one without a global VLAN-ID assigned to it.
    Such VLANs are outside the scope of IEEE 802.1Q, but it is convenient
    to be able to manage them in the same way using this YANG module.";
  reference
    "9.6 of IEEE Std 802.1Q";
}
typedef mstid-type {
  type uint32 {
    range "1..4094";
  }
  description
    "In an MSTP Bridge, an MSTID, i.e., a value used to identify a
    spanning tree (or MST) instance";
  reference
    "13.8 of IEEE Std 802.1Q";
}
typedef pcp-selection-type {
  type enumeration {
    enum 8P0D {
      description
        "8 priorities, 0 drop eligible";
    }
    enum 7P1D {
      description
        "7 priorities, 1 drop eligible";
    }
    enum 6P2D {
      description
        "6 priorities, 2 drop eligible";
    }
    enum 5P3D {
      description
        "5 priorities, 3 drop eligible";
    }
  }
  description
    "Priority Code Point selection types.";
  reference
    "12.6.2.5.3, 6.9.3 of IEEE Std 802.1Q";
}
typedef protocol-frame-format-type {
  type enumeration {
    enum Ethernet {
      description
        "Ethernet frame format";
    }
    enum rfc1042 {
      description
        "RFC 1042 frame format";
    }
    enum snap8021H {
```

```
      description
        "SNAP 802.1H frame format";
    }
    enum snapOther {
      description
        "Other SNAP frame format";
    }
    enum llcOther {
      description
        "Other LLC frame format";
    }
  }
  description
    "A value representing the frame format to be matched.";
  reference
    "12.10.1.7.1 of IEEE Std 802.1Q";
}
typedef ethertype-type {
  type string {
    pattern "[0-9a-fA-F]{2}-[0-9a-fA-F]{2}";
  }
  description
    "The EtherType value represented in the canonical order defined by
    IEEE 802. The canonical representation uses uppercase characters.";
  reference
    "9.2 of IEEE Std 802";
}
typedef dot1q-tag-type {
  type identityref {
    base dot1q-vlan-type;
  }
  description
    "Identifies a specific 802.1Q tag type";
  reference
    "9.5 IEEE Std 802.1Q";
}
typedef traffic-class-type {
  type uint8 {
    range "0..7";
  }
  description
    "This is the numerical value associated with a traffic class in a
    Bridge. Larger values are associated with higher priority traffic
    classes.";
  reference
    "3.273 of IEEE Std 802.1Q";
}
  typedef isid-type {
  type uint32 {
    range "1|255..16777214";
  }
  description
    "The i-sid type represents a Backbone Service Instance Identifier
    (I-SID). This is the 24-bit I-SID field used in the I-TAG TCI of a
    Provider Backbone Bridging frame. The values 0, 2-254, and 16777215
    are reserved for future standardization. The value 255 is dedicated
    to use by the SPB protocol.";
  reference
    "3.108, 9.7 of IEEE Std 802.1Q-2022";
}
grouping dot1q-tag-classifier-grouping {
  description
    "A grouping which represents an 802.1Q VLAN, matching both the
    EtherType and a single VLAN ID.";
  leaf tag-type {
    type dot1q-tag-type;
    mandatory true;
    description
      "VLAN type";
  }
  leaf vlan-id {
    type vlanid;
```

```
    mandatory true;
    description
      "VLAN ID";
  }
}
grouping dot1q-tag-or-any-classifier-grouping {
  description
    "A grouping which represents an 802.1Q VLAN, matching both the
    EtherType and a single VLAN ID or 'any' to match on any VLAN ID.";
  leaf tag-type {
    type dot1q-tag-type;
    mandatory true;
    description
      "VLAN type";
  }
  leaf vlan-id {
    type union {
      type vlanid;
      type enumeration {
        enum any {
          value 4095;
          description
            "Matches 'any' VLAN in the range 1 to 4094 that is not matched
            by a more specific VLAN ID match";
        }
      }
    }
    mandatory true;
    description
      "VLAN ID or any";
  }
}
grouping dot1q-tag-ranges-classifier-grouping {
  description
    "A grouping which represents an 802.1Q VLAN that matches a range of
    VLAN IDs.";
  leaf tag-type {
    type dot1q-tag-type;
    mandatory true;
    description
      "VLAN type";
  }
  leaf vlan-ids {
    type vid-range-type;
    mandatory true;
    description
      "VLAN IDs";
  }
}
grouping dot1q-tag-ranges-or-any-classifier-grouping {
  description
    "A grouping which represents an 802.1Q VLAN, matching both the
    EtherType and a single VLAN ID, ordered list of ranges, or 'any' to
    match on any VLAN ID.";
  leaf tag-type {
    type dot1q-tag-type;
    mandatory true;
    description
      "VLAN type";
  }
  leaf vlan-id {
    type union {
      type vid-range-type;
      type enumeration {
        enum any {
          value 4095;
          description
            "Matches 'any' VLAN in the range 1 to 4094.";
        }
      }
    }
    mandatory true;
```

```
    description
      "VLAN IDs or any";
  }
}
grouping priority-regeneration-table-grouping {
  description
    "The priority regeneration table provides the ability to map incoming
    priority values on a per-Port basis, under management control.";
  reference
    "6.9.4 of IEEE Std 802.1Q";
  leaf priority0 {
    type priority-type;
    default "0";
    description
      "Priority 0";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority1 {
    type priority-type;
    default "1";
    description
      "Priority 1";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority2 {
    type priority-type;
    default "2";
    description
      "Priority 2";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority3 {
    type priority-type;
    default "3";
    description
      "Priority 3";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority4 {
    type priority-type;
    default "4";
    description
      "Priority 4";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority5 {
    type priority-type;
    default "5";
    description
      "Priority 5";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority6 {
    type priority-type;
    default "6";
    description
      "Priority 6";
    reference
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
  leaf priority7 {
    type priority-type;
    default "7";
    description
      "Priority 7";
    reference
```

```
      "12.6.2.3, 6.9.4 of IEEE Std 802.1Q";
  }
}
grouping pcp-decoding-table-grouping {
  description
    "The Priority Code Point decoding table enables the decoding of the
    priority and drop-eligible parameters from the PCP.";
  reference
    "6.9.3 of IEEE Std 802.1Q";
  list pcp-decoding-map {
    key "pcp";
    description
      "This map associates the priority code point field found in the VLAN
      with a priority and drop eligible value based upon the priority code
      point selection type.";
    leaf pcp {
      type pcp-selection-type;
      description
        "The priority code point selection type.";
      reference
        "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
    }
    list priority-map {
      key "priority-code-point";
      description
        "This map associates a priority code point value with priority
        and drop eligible parameters.";
      leaf priority-code-point {
        type priority-type;
        description
          "Priority associated with the pcp.";
        reference
          "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
      }
      leaf priority {
        type priority-type;
        description
          "Priority associated with the pcp.";
        reference
          "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
      }
      leaf drop-eligible {
        type boolean;
        description
          "Drop eligible value for pcp";
        reference
          "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
      }
    }
  }
}
grouping pcp-encoding-table-grouping {
  description
    "The Priority Code Point encoding table encodes the priority and
    drop-eligible parameters in the PCP field of the VLAN tag.";
  reference
    "12.6.2.9, 6.9.3 of IEEE Std 802.1Q";
  list pcp-encoding-map {
    key "pcp";
    description
      "This map associates the priority and drop-eligible parameters
      with the priority used to encode the PCP of the VLAN based upon
      the priority code point selection type.";
    leaf pcp {
      type pcp-selection-type;
      description
        "The priority code point selection type.";
      reference
        "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
    }
    list priority-map {
      key "priority dei";
```

```
      description
        "This map associates the priority and drop-eligible parameters
        with the priority code point field of the VLAN tag.";
      leaf priority {
        type priority-type;
        description
          "Priority associated with the pcp.";
        reference
          "12.6.2.7, 6.9.3 of IEEE Std 802.1Q";
      }
      leaf dei {
        type boolean;
        description
          "The drop eligible value.";
        reference
          "12.6.2, 8.6.6 of IEEE Std 802.1Q";
      }
      leaf priority-code-point {
        type priority-type;
        description
          "PCP value for priority when DEI value";
        reference
          "12.6.2.9, 6.9.3 of IEEE Std 802.1Q";
      }
    }
  }
}
grouping service-access-priority-table-grouping {
  description
    "The Service Access Priority Table associates a received priority with
    a service access priority.";
  reference
    "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
  leaf priority0 {
    type priority-type;
    default "0";
    description
      "Service access priority value for priority 0";
    reference
      "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
  }
  leaf priority1 {
    type priority-type;
    default "1";
    description
      "Service access priority value for priority 1";
    reference
      "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
  }
  leaf priority2 {
    type priority-type;
    default "2";
    description
      "Service access priority value for priority 2";
    reference
      "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
  }
  leaf priority3 {
    type priority-type;
    default "3";
    description
      "Service access priority value for priority 3";
    reference
      "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
  }
  leaf priority4 {
    type priority-type;
    default "4";
    description
      "Service access priority value for priority 4";
    reference
      "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
```

```
    }
    leaf priority5 {
      type priority-type;
      default "5";
      description
        "Service access priority value for priority 5";
      reference
        "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
    }
    leaf priority6 {
      type priority-type;
      default "6";
      description
        "Service access priority value for priority 6";
      reference
        "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
    }
    leaf priority7 {
      type priority-type;
      default "7";
      description
        "Service access priority value for priority 7";
      reference
        "12.6.2.17, 6.13.1 of IEEE Std 802.1Q";
    }
  }
  grouping traffic-class-table-grouping {
    description
      "The Traffic Class Table models the operations that can be
      performed on, or can inquire about, the current contents of the
      Traffic Class Table (8.6.6) for a given Port.";
    reference
      "12.6.3, 8.6.6 of IEEE Std 802.1Q";
    list traffic-class-map {
      key "priority";
      status "deprecated";
      description
        "The priority index into the traffic class table. This list modeled
        the Traffic Class Table incorrectly. available-traffic-class should
        be a single value per port and not a list of all possible
        available-traffic-class. For more information see maintenance issue
        0230. The status of this object is deprecated. It is replaced by
        sibling container traffic-class-table.";
      leaf priority {
        type priority-type;
        description
          "The priority of the traffic class entry.";
        reference
          "8.6.6 of IEEE Std 802.1Q";
      }
      list available-traffic-class {
        key "num-traffic-class";
        description
          "The traffic class index associated with a given priority within
          the traffic class table.";
        reference
          "8.6.6 of IEEE Std 802.1Q";
        leaf num-traffic-class {
          type uint8 {
            range "1..8";
          }
          description
            "The available number of traffic classes.";
          reference
            "8.6.6 of IEEE Std 802.1Q";
        }
        leaf traffic-class {
          type traffic-class-type;
          description
            "The traffic class index associated with a given traffic class
            entry.";
          reference
```

```
        "8.6.6 of IEEE Std 802.1Q";
    }
  }
}
container traffic-class-table {
  description
    "The priority to traffic class mapping.";
  reference
    "8.6.6 of IEEE Std 802.1Q";
  leaf number-of-traffic-classes {
    type uint8 {
      range "1..8";
    }
    description
      "The number of egress traffic classes supported on this port. This
      object may be optionally read-only.";
    reference
      "12.6.3.1 of IEEE Std 802.1Q";
  }
  leaf priority0 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 0";
    reference
      "8.6.6 of IEEE Std 802.1Q";
  }
  leaf priority1 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 1";
    reference
      "8.6.6 of IEEE Std 802.1Q";
  }
  leaf priority2 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 2";
    reference
      "8.6.6 of IEEE Std 802.1Q";
  }
  leaf priority3 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 3";
    reference
      "8.6.6 of IEEE Std 802.1Q";
  }
  leaf priority4 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 4";
    reference
      "8.6.6 of IEEE Std 802.1Q";
  }
  leaf priority5 {
    type traffic-class-type;
    must
      "current() < ../number-of-traffic-classes";
    description
      "The traffic class index associated with priority 5";
    reference
      "8.6.6 of IEEE Std 802.1Q";
```

```
    }
    leaf priority6 {
      type traffic-class-type;
      must
        "current() < ../number-of-traffic-classes";
      description
        "The traffic class index associated with priority 6";
      reference
        "8.6.6 of IEEE Std 802.1Q";
    }
    leaf priority7 {
      type traffic-class-type;
      must
        "current() < ../number-of-traffic-classes";
      description
        "The traffic class index associated with priority 7";
      reference
        "8.6.6 of IEEE Std 802.1Q";
    }
  }
}
grouping transmission-selection-table-grouping {
  description
    "The Transmission Selection Algorithm Table models the operations that
    can be performed on, or can inquire about, the current contents of the
    Transmission Selection Algorithm Table (12.20.2) for a given Port.";
  reference
    "12.20.2, 8.6.8 of IEEE Std 802.1Q";
  list transmission-selection-algorithm-map {
    key "traffic-class";
    description
      "The traffic class to index into the transmission selection table.";
    leaf traffic-class {
      type traffic-class-type;
      description
        "The traffic class of the entry.";
      reference
        "8.6.6 of IEEE Std 802.1Q";
    }
    leaf transmission-selection-algorithm {
      type identityref {
        base dot1q-types:transmission-selection-algorithm;
      }
      description
        "Transmission selection algorithm";
      reference
        "8.6.8, Table 8-6 of IEEE Std 802.1Q";
    }
  }
}
grouping port-map-grouping {
  description
    "A set of control indicators, one for each Port. A Port Map,
    containing a control element for each outbound Port";
  reference
    "8.8.1, 8.8.2 of IEEE Std 802.1Q";
  list port-map {
    key "port-ref";
    description
      "The list of entries composing the port map.";
    leaf port-ref {
      type port-number-type;
      description
        "The interface port reference associated with this map.";
      reference
        "8.8.1 of IEEE Std 802.1Q";
    }
    choice map-type {
      description
        "Type of port map";
      container static-filtering-entries {
        description
```

```
      "Static filtering entries attributes.";
    leaf control-element {
      type enumeration {
        enum forward {
          description
            "Forwarded, independently of any dynamic filtering
            information held by the FDB.";
        }
        enum filter {
          description
            "Filtered, independently of any dynamic filtering
            information.";
        }
        enum forward-filter {
          description
            "Forwarded or filtered on the basis of dynamic filtering
            information, or on the basis of the default Group
            filtering behavior for the outbound Port (8.8.6) if no
            dynamic filtering information is present specifically for
            the MAC address.";
        }
      }
      description
        "A control element for each outbound Port, specifying that
        a frame with a destination MAC address, and in the case of
        VLAN Bridge components, VID that meets this specification.";
      reference
        "8.8.1 of IEEE Std 802.1Q";
    }
    leaf connection-identifier {
      type port-number-type;
      description
        "A Port MAP may contain a connection identifier (8.8.12) for
        each outbound port. The connection identifier may be
        associated with the Bridge Port value maintained in a Dynamic
        Filtering Entry of the FDB for Bridge Ports.";
      reference
        "8.8.1, 8.8.12 of IEEE Std 802.1Q";
    }
  }
  container static-vlan-registration-entries {
    description
      "Static VLAN registration entries.";
    leaf registrar-admin-control {
      type enumeration {
        enum fixed-new-ignored {
          description
            "Registration Fixed (New ignored).";
        }
        enum fixed-new-propagated {
          description
            "Registration Fixed (New propagated.";
        }
        enum forbidden {
          description
            "Registration Forbidden.";
        }
        enum normal {
          description
            "Normal Registration.";
        }
      }
      description
        "The Registrar Administrative Control values for MVRP and MIRP
        for the VID.";
      reference
        "8.8.2 of IEEE Std 802.1Q";
    }
    leaf vlan-transmitted {
      type enumeration {
        enum tagged {
          description
```

```
            "VLAN-tagged";
        }
        enum untagged {
          description
            "VLAN-untagged";
        }
      }
      description
        "Whether frames are to be VLAN-tagged or untagged when
        transmitted.";
      reference
        "8.8.2 of IEEE Std 802.1Q";
    }
  }
  container mac-address-registration-entries {
    description
      "MAC address registration entries attributes.";
    leaf control-element {
      type enumeration {
        enum registered {
          description
            "Forwarded, independently of any dynamic filtering
            information held by the FDB.";
        }
        enum not-registered {
          description
            "Filtered, independently of any dynamic filtering
            information.";
        }
      }
      description
        "A control element for each outbound Port, specifying that
        a frame with a destination MAC address, and in the case of
        VLAN Bridge components, VID that meets this specification.";
      reference
        "8.8.4 of IEEE Std 802.1Q";
    }
  }
  container dynamic-vlan-registration-entries {
    description
      "Dynamic VLAN registration entries attributes.";
    leaf control-element {
      type enumeration {
        enum registered {
          description
            "Forwarded, independently of any dynamic filtering
            information held by the FDB.";
        }
      }
      description
        "A control element for each outbound Port, specifying that
        a frame with a destination MAC address, and in the case of
        VLAN Bridge components, VID that meets this specification.";
      reference
        "8.8.5 of IEEE Std 802.1Q";
    }
  }
  container dynamic-reservation-entries {
    description
      "Dynamic reservation entries attributes.";
    leaf control-element {
      type enumeration {
        enum forward {
          description
            "Forwarded, independently of any dynamic filtering
            information held by the FDB.";
        }
        enum filter {
          description
            "Filtered, independently of any dynamic filtering
            information.";
        }
```

```
          }
          description
            "A control element for each outbound Port, specifying that
            a frame with a destination MAC address, and in the case of
            VLAN Bridge components, VID that meets this specification.";
          reference
            "8.8.7 of IEEE Std 802.1Q";
        }
      }
      container dynamic-filtering-entries {
        description
          "Dynamic filtering entries attributes.";
        leaf control-element {
          type enumeration {
            enum forward {
              description
                "Forwarded, independently of any dynamic filtering
                information held by the FDB.";
            }
          }
          description
            "A control element for each outbound Port, specifying that
            a frame with a destination MAC address, and in the case of
            VLAN Bridge components, VID that meets this specification.";
          reference
            "8.8.3 of IEEE Std 802.1Q";
        }
      }
    }
  }
}
grouping bridge-port-statistics-grouping {
  description
    "Grouping of bridge port statistics.";
  reference
    "12.6.1.1.3 of IEEE Std 802.1Q";
  leaf delay-exceeded-discards {
    type yang:counter64;
    description
      "The number of frames discarded by this port due to excessive
      transit delay through the Bridge. It is incremented by both
      transparent and source route Bridges.";
    reference
      "12.6.1.1.3, 8.6.6 of IEEE Std 802.1Q";
  }
  leaf mtu-exceeded-discards {
    type yang:counter64;
    description
      "The number of frames discarded by this port due to an excessive
      size. It is incremented by both transparent and source route
      Bridges.";
    reference
      "Item g) in 12.6.1.1.3 of IEEE Std 802.1Q";
  }
  leaf frame-rx {
    type yang:counter64;
    description
      "The number of frames that have been received by this port from its
      segment. Note that a frame received on the interface corresponding
      to this port is only counted by this object if and only if it is for
      a protocol being processed by the local bridging function, including
      Bridge management frames.";
    reference
      "12.6.1.1.3 of IEEE Std 802.1Q";
  }
  leaf octets-rx {
    type yang:counter64;
    description
      "The total number of octets in all valid frames received (including
      BPDUs, frames addressed to the Bridge as an end station, and frames
      that were submitted to the Forwarding Process).";
    reference
```

```
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
    leaf frame-tx {
      type yang:counter64;
      description
        "The number of frames that have been transmitted by this port to its
        segment. Note that a frame transmitted on the interface
        corresponding to this port is only counted by this object if and
        only if it is for a protocol being processed by the local bridging
        function, including Bridge management frames.";
    }
    leaf octets-tx {
      type yang:counter64;
      description
        "The total number of octets that have been transmitted by this port
        to its segment.";
    }
    leaf discard-inbound {
      type yang:counter64;
      description
        "Count of received valid frames that were discarded (i.e., filtered)
        by the Forwarding Process.";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
    leaf forward-outbound {
      type yang:counter64;
      description
        "The number of frames forwarded to the associated MAC Entity (8.5).";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
    leaf discard-lack-of-buffers {
      type yang:counter64;
      description
        "The count of frames that were to be transmitted through the
        associated Port but were discarded due to lack of buffers.";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
    leaf discard-transit-delay-exceeded {
      type yang:counter64;
      description
        "The number of frames discarded by this port due to excessive
        transit delay through the Bridge. It is incremented by both
        transparent and source route Bridges.";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
    leaf discard-on-error {
      type yang:counter64;
      description
        "The number of frames that were to be forwarded on the associated
        MAC but could not be transmitted (e.g., frame would be too large,
        6.5.8).";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q";
    }
  }
  grouping base-gate-control-entries {
    description
      "Grouping of gate control entries.";
    list gate-control-entry {
      key "index";
      description
        "List of gate control entries.";
      leaf index {
        type uint32;
        description
          "The index for this entry.";
      }
      leaf operation-name {
```

```
        type identityref {
          base type-of-operation;
        }
        mandatory true;
        description
          "The name (type) of the operation for this entry.";
      }
      leaf time-interval-value {
        type uint32;
        description
          "timeIntervalValue is a 32-bit unsigned integer, representing a
          number of nanoseconds. After timeIntervalValue nanoseconds have
          elapsed since the completion of the previous entry in the gate
          control list, control passes to the next entry.";
        reference
          "12.29.1.2.3, 8.6.8.4 of IEEE Std 802.1Q";
      }
    }
  }
}
```

Insert Clause 50 after Clause 49 (inserted by IEEE Std 802.1Qcz-2023), as follows:

50. PBBN Auto Attach

When a C-VLAN or S-VLAN aware system is connected to a Provider Backbone Bridged Network (PBBN), the Backbone Edge Bridge's (BEB's) mapping of the Backbone Service Instance Identifiers (I-SIDs) to the C-VLAN or S-VLAN Identifiers (C-VIDs or S-VIDs) used by the system needs to be configured. This clause specifies an Auto Attach Protocol (AAP) that runs over the Link Layer Discovery Protocol (LLDP, IEEE Std 802.1AB) and allows Auto Attach capable devices (AADs) and BEBs (AABs) to automate this configuration. Clause 12 (Bridge Management) and Clause 17 (Management Information Base) specify supporting management functionality, and Annex D specifies LLDP TLVs and TLV management for PBBN Auto Attach.

50.1 Overview

In order to automate the attachment of Auto Attach Devices (AADs) to an Auto Attach BEB (AAB), the Auto Attach Protocol (AAP) is used. The AAP, which runs over LLDP, avoids manual configuration by mapping C-VIDs or S-VIDs to I-SIDs to establish a C-VLAN or S-VLAN to BSI mapping at the AAB. The AAP is only supported over a point-to-point single peer-to-peer topology.

Figure 50-1 depicts the Auto Attach architectural framework model that allows an AAD to communicate VLAN to BSI mapping requests for attachment over a PBBN.

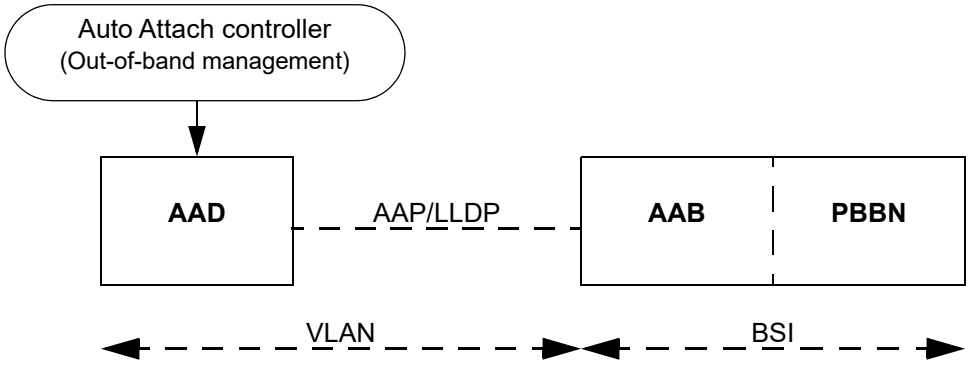

Figure 50-1—PBBN Auto Attach model

The conceptual example in Figure 50-1 depicts the process where an AAD end station or Bridge uses AAP running over LLDP to communicate the need to connect a C-VLAN to the appropriate BSI via the AAB. LLDP agents exchange IEEE 802.1 Organizationally Specific TLVs (D.2.17, D.2.18) to automate VID to I-SID assignment and registration.

An AAD is an end station, C-VLAN Bridge, S-VLAN Bridge, or MAC Bridge that advertises desired C-VID or S-VID to I-SID assignments using the PBBN Auto Attach System TLV (D.2.17) and PBBN Auto Attach Assignment TLV (D.2.18).

An AAB is a BEB along with C-VLAN components that manages and maintains C-VID or S-VID to I-SID assignments used to attach AADs. An AAB uses Auto Attach System and Auto Attach Assignment TLVs to activate C-VID or S-VID to I-SID assignments advertised by the Auto Attach Assignment TLVs.

The AAP operates between an AAB and an AAD port. The AAP controls the exchange of Auto Attach TLVs, and configuration of VLAN and BSI mappings between the participating systems.

The AAP invokes LLDP when there is a change to an Auto Attach TLV by invoking the SomethingChangedLocal() procedure, specified by IEEE Std 802.1AB, causing transmission to a peer AAS. Likewise, an AAS is invoked by LLDP when there is an update to an Auto Attach TLV triggering the SomethingChangedRemote() procedure, specified by IEEE Std 802.1AB.

Figure 50-2 depicts the Auto Attach functions that invoke the base LLDP transmission and reception of LLDPDUs. The LLDP state machines are defined in Clause 9 of IEEE Std 802.1AB-2016.

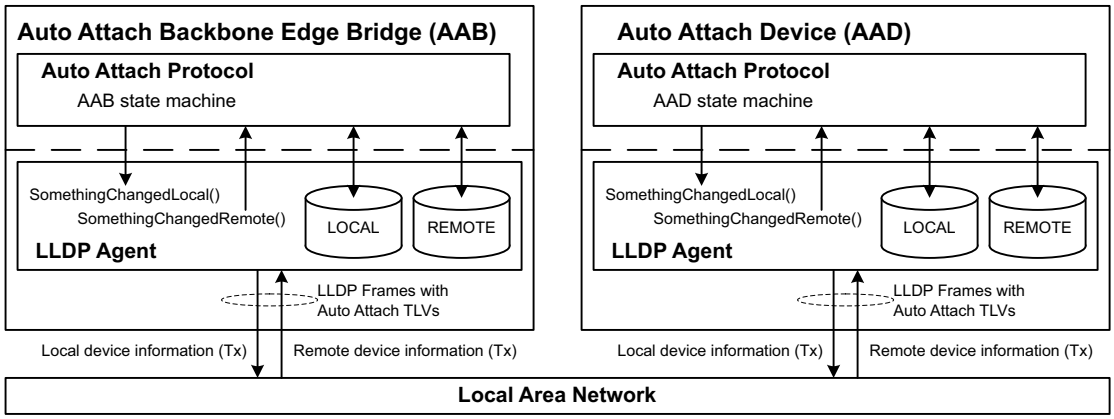

Figure 50-2—PBBN Auto Attach functions diagram

50.2 Service interfaces

An AAB provides two types of service interface, one for S-tagged service and another for C-tagged service. Which of these interfaces is selected is determined by the AAD system type requesting attachment. If the AAD system type is S-VLAN aware AAD, then the AAB provides an S-tagged service interface and all mappings within the assignment TLV are for S-VID to I-SID. For all other AAD types, a C-tagged service interface is provided and all mappings within the assignment TLV are for C-VID to I-SID.

50.2.1 S-tagged service interface

The S-tagged service interface maps a service instance from a Provider Bridged Network (PBN), identified by an S-VID, to a backbone service instance on the PBBN, identified by an I-SID. The S-tagged service interface performs a one-to-one mapping of S-VIDs to I-SIDs. Frames that are mapped to the I-SID are carried over the PBBN while frames that are not mapped to an I-SID are not carried over the PBBN.

An S-tagged service interface is provided by an AAB over a Customer Network Port provided by the AAB's I-component. The S-tagged interface does not carry the S-TAG over the PBBN. The DEI and PCP bits are regenerated on ingress and are then carried in the I-DEI and I-PCP bits in the I-TAG across the PBBN. On egress to an S-tagged interface, the S-TAG can be deduced from the I-TAG received from the PBBN (the I-SID is mapped to an S-VID, the I-DEI and I-PCP bits are regenerated and then carried in the DEI and PCP bits).

50.2.2 C-tagged service interface

The C-tagged service interface maps a service instance from a Customer Bridged Network (CBN), identified by a C-VID, to a backbone service instance on the PBBN, identified by an I-SID. The C-tagged service interface performs a one-to-one mapping of C-VIDs to I-SIDs. Frames that are mapped to the I-SID are carried over the PBBN while frames that are not mapped to an I-SID are not carried over the PBBN.

A C-tagged service interface is provided by an AAB over a Customer Edge Port (CEP) as illustrated by Figure 50-3. The C-tagged interface does not carry the C-TAG over the PBBN. The DEI and PCP bits are regenerated on ingress and are then carried in the I-DEI and I-PCP bits in the I-TAG across the PBBN. On egress to a C-tagged interface, the C-TAG can be deduced from the I-TAG received from the PBBN (the I-SID is mapped to a C-VID, the I-DEI and I-PCP bits are regenerated and then carried in the DEI and PCP bits).

Each of the C-VLAN components used to provide CEP service at the edge of an AAB is composed of a single CEP and a single Provider Edge Port (PEP) for each service instance that can be provided through that CEP. Each PEP is connected within the AAB, as specified in 6.14, to a distinct CNP on the I-component. Each C-VLAN component implements RSTP, with the enhancements to support CEPs, as specified in 13.41.

NOTE 1—The restriction that each CBN C-VLAN map to a single backbone service instance on the PBBN allows the CBN equipment receiving frames to correctly identify the service instance used to deliver that frame and prevents the configuration of the I-component to create a multi-point service from point-to-point service instances, which could result in the creation of data loops. The backbone provider can offer a multi-point service through appropriate configuration of the B-VLAN component.

Figure 50-3 illustrates a CBN attached to a PBBN using a C-tagged service interface. The customer network uses C-VLAN aware end stations or Bridges connecting to the PBBN. The PBBN in turn is composed of AABs or BEBs interfacing to the customer Provider Bridges and BCBs used to forward frames between the AABs or BEBs.

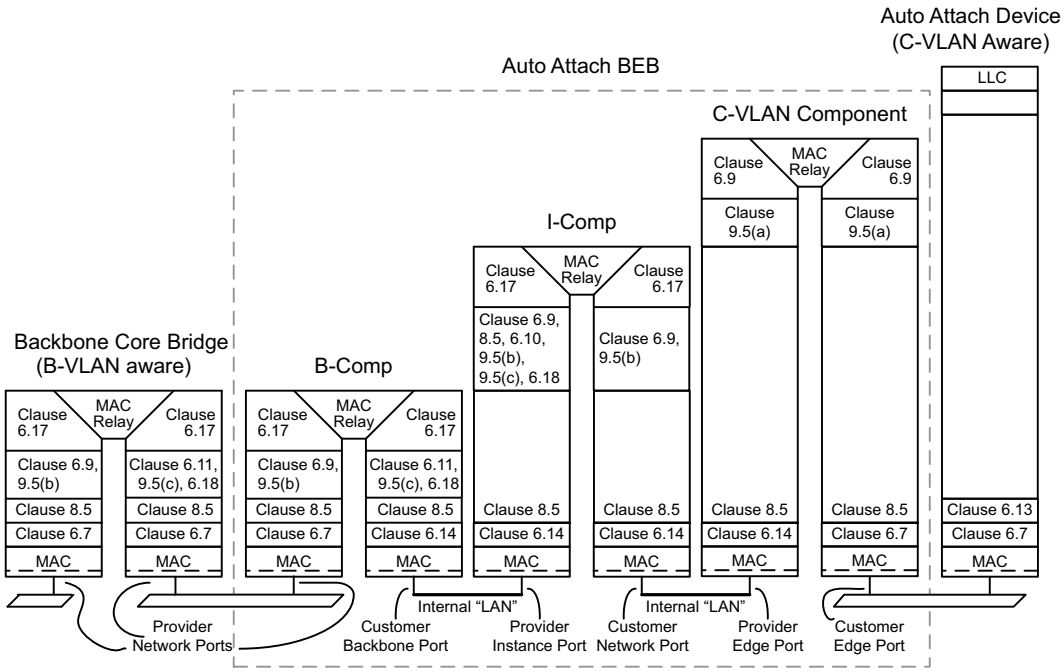

Figure 50-3—PBBN Auto Attach for customer networks

Customer VLAN unaware Bridges and MAC relay end station equipment can connect to Customer Network Ports (CNPs) using an unprotected port-based service interface (25.3).

Service attachment requests from VLAN unaware (untagged) AADs send an assignment TLV with one VID/I-SID assignment to the AAB. The port-based service interface on the AAB uses the specified VID to update port VLAN membership used by the CNP to map the VID to an I-component.

A Port-based service interface allows a single mapping instance of an untagged service to an I-SID.

NOTE 2—An AAB includes both BEB and Provider Edge Bridge components. In particular, an AAB supports a C-tagged service interface (15.4) for attachment to a CBN by including a C-VLAN component with PEPs that connect to the CNPs of an I-component.

50.3 State machine overview

The AAP is specified by the external behavior indicated by the AAD state machine in 50.6 and the AAB state machine in 50.7. These AAP state machines operate over LLDP on a point-to-point single peer-to-peer topology. Each port enabled for PBBN Auto Attach runs an independent copy of the AAP.

When Auto Attach is enabled on a port, the AAP begins to write and read information to the local system LLDP database. The initial process for an AAS is to write system information into the local LLDP database. Information written into the LLDP database is transmitted on the port within the PBBN Auto Attach System TLV in the next LLDPDU interval. Once an AAS has learned its neighbors and formed an association with a neighboring AAS, the AAD can update the local LLDP database with assignment mappings that are transmitted on the port within the PBBN Auto Attach Assignment TLV. The AAB receives the assignment mappings advertised by the AAD in the remote LLDP database. When the AAB receives new VID/I-SID assignments from the AAD it determines if it can activate them. The AAB then updates its local database with the VID/I-SID pairs from the AAD and indicates the status as pending, active, or rejected-with reasons. For rejected assignments the AAB supplies some further information on the cause for the rejection.

The AAP state machine diagrams (50.6, 50.7) specify state transitions based on changes to the local structures aaRemSysTlv and aaRemAsgnsTlv. These structures are updated whenever the LLDP routine SomethingChangedRemote() calls the Auto Attach LLDP call back procedure (50.5.16). The structure aaRemSysTlv reflects the current PBBN Auto Attach System TLV within the remote LLDP database, however transformed to machine local byte and bit order. The structure aaRemAsgnsTlv reflects the current PBBN Auto Attach Assignment TLV within the remote LLDP database, however transformed to machine local byte and bit order. If no remote database is present, or if either Auto Attach TLV is absent in the remote database, then aaRemSysTlv and(or) aaRemAsgnsTlv is set to NULL.

When a system TLV (D.2.17) is received, the AAS enters the VALIDATE_REMOTE_ASSOCIATION state to determine whether an association can be formed with the AAS neighbor.

The following actions are performed by the state machine to validate an association:

- For an AAS verify this port has a point-to-point single peer-to-peer topology on this port by verifying there is only a single remote LLDP database
- For an AAS verify there is only a single system TLV in the remote database
- For an AAD verify the assoc state of the remote system is READY-TO-ASSOC
- For an AAD verify the remote system type is AAB
- For an AAD verify the AAB tagging matches the AAD
- For an AAB verify the remote system type is AAD
- For an AAB verify the AAD tagging is supported and indicated in the AAB system TLV

If there is an error validating the remote system, such as a system type mismatch, VALIDATE_REMOTE_ASSOCIATION continues to check any received system TLVs until an error free configuration is detected.

Once the system TLV validation has completed, the AAD verifies if it already has an active association on a different port. If no other association is active, then the AAD activates the association on this port and begins sending assignment TLVs. If the AAD finds it already has an active association on a different port it holds this association in standby and does not issue an assignment TLV for this port. If the AAD active association is disabled or fails, the AAD state machine resets the port and restarts the association process on that port. On restart the AAD breaks all assignments on the reset port, withdraws all Auto Attach TLVs, resynchronizes with the peer AAB state machine, and starts the association process over. If the AAD has any associations in standby, the first to recognize the active association has broken becomes active and issues an assignment TLV.

The AAB can only accept assignment TLVs from an AAD with a valid system TLV. The AAB only allows a single active association for each AAD. If the AAB finds an AAD attempting to activate a port with an invalid system TLV or if the AAD attempts to activate more than one association, the AAB resets its port invalidating the association and forcing a restart of the peer AAD state machine. On restart the AAB breaks all assignments on the reset port, withdraws all Auto Attach TLVs, resynchronizes with the peer AAD state machine, and starts the association process over.

An AAD is responsible for initiating one or more VID/I-SID assignment binding requests and sending them to the AAB in the assignment TLV. The AAD uses aaAdminPortVidIsidAsgns[] (50.4.3.2) to determine the requested assignments. Bindings are individually processed by the AAB to determine whether the specified VID can be bound to the I-SID. If the binding is allowed, the AAB creates the VLAN and BSI as indicated by the AAD assignment TLV. Once all assignments are updated, the assignment TLV is updated by the AAB and sent to the AAD with the status of all assignment requests. The AAD updates all VID/I-SID assignments and reflects the current status back to the AAB in its assignment TLV.

50.3.1 System inactivity

If the remote LLDP database for an AAS is deleted, Auto Attach TLVs are deleted from the database, or Auto Attach System TLVs are not received, then all operating VID to I-SID assignments are deactivated and the AAP restarts.

50.3.2 Multiple AAD systems

The Auto Attach function is intended to operate in a point-to-point single peer-to-peer topology. Multiple AADs are not permitted to form an association on a single AAB port. Only one AAD system is permitted to form an association per port interface of an AAB.

50.3.3 AAD assignment requests

Assignment mapping requests are communicated from the AAD to the AAB using the PBBN Auto Attach Assignment TLV (D.2.18). These service assignment requests are a combination of the VID/I-SID mapping and assignment status.

Assignment mappings are established when two criteria are met:

a) An AAB is discovered by an AAD and has a valid remote association determined by the ValidateRemoteAssociation state machine procedures (50.6).
b) The Auto Attach variable aaAdminPortVidIsidAsgns[] for this port is configured with one or more VID/I-SID assignments using the management interface (12.34.4).

50.3.4 AAB assignment request processing

Each VLAN that is associated with a VID/I-SID assignment mapping is configured on the AAD by setting the VID/I-SID mapping in the PBBN Auto Attach assignment table (12.34.4). The port associated with the link connecting the AAD to the AAB is a member of the VLAN(s) in the VID/I-SID assignment list received and accepted by the AAB. This allows traffic on these VLANs to pass into the BSI.

All initial VID/I-SID assignment mapping requests are sent by the AAD with an assignment status of pending until the AAB completes processing the request. The AAD updates and reflects the assignment status changes for each VID/I-SID according to the last assignment status received from the AAB.

Each VID/I-SID assignment request received by the AAB is processed individually and is either accepted or rejected. Assignment requests are processed for interfaces on which the port is in operation, LLDP is enabled, Auto Attach TLV processing is enabled, and AAS valid association has occurred.

If an assignment is accepted the following happens:

— Each VLAN that is associated with an accepted VID/I-SID assignment is instantiated on the AAB if it does not already exist, in order that proper data packet encapsulation and decapsulation is supported.

There are two conditions associated with the interface assignment mapping and binding process:

a) If the I-SID already exists on the AAB, due to a previous configuration from any type of interface binding method, then the AAB configures the associated interface to the pre-existing I-SID.
b) If the I-SID does not exist on the AAB already, then it is responsible for creating the BSI with the associated I-SID.

Port tagging and port VLAN membership updates are performed by the AAD based on assignment acceptance.

If an assignment is rejected, a response with a reason is returned to the AAD based on the following:

— Generic (undefined rejection error)
— Auto Attach resources unavailable
— Invalid VLAN ID
— VLAN resources unavailable
— Invalid I-SID
— I-SID resources unavailable
— Application interaction (VLAN, PBB) issue
— Assignment not allowed

Rejected assignments are returned to the originating AAD with a rejected status and indicate why the rejection occurred using status as defined in D.2.18.6. An AAD continues to request a rejected assignment until the assignment is administratively removed.

50.3.5 Assignment updates

All active VID/I-SID assignments on the AAB are updated using the current assignment TLV information from the AAD.

When a new assignment TLV is received at the AAB containing additional mapping requests, removing existing mapping requests, or changing existing mapping requests, the AAB state machine processes the new requests by adding, deleting, or updating the assignments that were changed in the new assignment TLV.

If one or more VID/I-SID assignments are not present in any subsequent assignment TLVs from the AAD, the AAB removes the mapping of those missing assignments from the active list and releases the binding.

50.4 State machine variables

The Auto Attach state machines use both port local (50.4.3) and system wide variables (50.4.2). Updating a system wide variable requires assuring consistency since multiple instances of the state machines (one per port) can access the system wide variables.

The Auto Attach state machines use some complex structures (50.4.1). These structures are copied and compared within the state machines in the same manner as strings. Copies result in a new structure of the same length and content duplicated in a new variable. Compares test equal when two structures are exactly the same size and content, otherwise they are not equal.

Arrays are indexed with their first element starting at zero.

When manipulating structures the state machine diagrams use dot notation to illustrate access to objects within a structure. The syntax for accessing the objects within a structure used in the state machines is:

```
<structure instance>.<object name>
```

At times the state machine diagrams need to exclude a field from a structure that is being compared as a string. This is expressed by prefixing the object name with “except”:

```
<structure instance>.<except><object name>
```

50.4.1 Structure type definitions

Type definitions are used to define some of the state machine variables. They are represented by C style typedefs. Though C structures can not be manipulated and compared as though they are strings the state machine both copies and compares these structures to simplify the state machine description.

50.4.1.1 Assignment TLV type definition

The type *ASGNSstring* models the PBBN Auto Attach Assignment TLV (D.2.18). It contains elements for all the fields defined for the assignment TLV, however transformed into machine local byte and bit order.

Part of the *ASGNSsting* type is an array of triples, ASGNStriple, identifying the assignment status/VID/I-SID. This array is variable in size and so instances of ASGNSstring can vary in size.

```
typedef struct {
       Unsigned            status : 4;
       Unsigned            vid : 12;
       Unsigned            isid : 24;
} ASGNStriple;
typedef struct {
       Unsigned Char       numAsgns;
       ASGNStriple         asgnsArray[];
} ASGNSstring;
```

50.4.1.2 Discovered association table object

This object is the structure used by an AAS state machine for each object of the discovered remote attach port table (50.4.2.2).

```
typedef struct {
       Unsigned            locPortId;
       PORTNETIDstring     remPortNetId;
} DISCport;
```

50.4.1.3 System TLV type definition

The type *SYSstring* models the PBBN Auto Attach System TLV (D.2.17). It contains elements for all of the fields defined for the system TLV, however transformed into machine local byte and bit order.

```
typedef struct {
       Unsigned            sysMAC : 48;
       Unsigned16          reserved;
       Unsigned32          portId;
} PORTNETIDstring;
typedef struct {
       Unsigned char       state;
       Unsigned            sysType : 4;
       Unsigned            portTagging : 4;
       Unsigned char       reserved;
       PORTNETIDstring     portNetId;
} SYSstring;
```

50.4.1.4 VID to I-SID assignment table object

This object is the 2-tuples *VIDISIDpair* containing a single VID to I-SID assignment. This type is used by the state machine for typing arrays of the assignment requests.

```
typedef struct {
       Unsigned     vid : 12;
       Unsigned     isid : 24;
} VIDISIDpair;
```

50.4.2 Per AAS variables

50.4.2.1 aaAdminSystemEnable

When FALSE this Boolean variable causes all AAP state machines on all ports to immediately enter the INIT state. When set to TRUE this variable allows the AAP state machines to progress. The variable aaAdminSystemEnable is set by the management system object aaSystemEnable (12.34.1).

50.4.2.2 aaOperDiscAssocTable[]

This table lists all of the currently associated ports discovered on remote AASs. It is an array of type DISCport (50.4.1.2). When accessing or updating this table the state machine's implementation takes care of any required interlocking for sharing this table.

50.4.2.3 aaOperAadActiveAssocIndex

This variable provides an index into the aaOperDiscAssocTable[] for the active association on this AAD. If no association is active, then aaOperAadActiveAssocIndex is set to NO_ACTIVE_ASSOC. When updating or accessing this variable the state machine's implementation takes care of any required interlocking for sharing this variable.

50.4.2.4 aaSystemMAC

This variable is a MAC address selected to uniquely identify this AAS over the administrative domain. This variable is set by the system to an appropriate value (such as the LLDP ChassisID) at system startup. If the system is a DRNI the single aaSystemMAC refers to both devices in the DRNI pair.

50.4.2.5 aaSystemType

The system type can take the values (1)AAB, (2)CVLAN_AWARE_AAD, (3)VLAN_UNAWARE_AAD, or (4)SVLAN_AWARE_AAD. This variable is set by the system to an appropriate value at system startup.

50.4.2.6 aaSystemResetTime

This variable is used to initialize the aaResetTimer during a state machine reset. The time in seconds can range from 1 to 1200 seconds.

50.4.2.7 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.

50.4.3 Per port variables

50.4.3.1 aaAdminPortEnable

When FALSE this Boolean variable causes the AAP state machine on this port to immediately enter the INIT state. A value of TRUE allows the AAP state machine on the port to progress. The aaAdminPortEnable is set by the management port table object aaPortEnable (12.34.2).

50.4.3.2 aaAdminVidIsidAsgns[]

This is an array containing the administratively assigned VID/I-SID pairs desired by an AAD on this port. The array is loaded by management with the assignments table entries (12.34.4) for this port. This array is only used by the AAD state machine.

This variable is an array of 2-tuples each containing a VID to I-SID assignment of type VIDISIDpair (50.4.1.4). The state machine can compare or copy the entire array to an array of the same type with an assignment, compare equal, or compare not equal operator. To be equal the two arrays must be of the same size and contain the same elements in each 2-tuple.

The variables aaAdminVidIsidAsgns[n].vid and aaAdminVidIsidAsgns[n].isid of each 2-tuple within aaAdminVidIsidAsgns[] is exposed at the management interface. Each row of the management database assignment table represents an index into aaAdminVidIsidAsgns[]. So, for instance, the first row refers to aaAdminVidIsidAsgns[0] and value of the first VID/ISID pair is set to the objects of the first row of the Auto Attach assignment table entry for this port.

50.4.3.3 aaAsgnsTlvEnable

This variable enables and disables transmission of the PBBN Auto Attach Assignment TLV on this port. This variable is the LLDP MIB object lldpXdot1AaConfigAsgnsTxEnable (D.5.6) or LLDP YANG object tlvs-tx-org-aa-system-enable.aaAssign (D.6.6.7).

PBBN Auto Attach assignment TLV transmission is controlled by the Auto Attach state machines. When aaAsgnsTlvEnable is FALSE, LLDP does not include the Auto Attach Assignment TLV in its transmissions. When the aaAsgnsTlvEnable is TRUE, LLDP includes the Auto Attach Assignment TLV in its transmissions.

50.4.3.4 aaCurVidIsidAsgns[]

This variable is an array of 2-tuples each containing a VID to I-SID assignment of type VIDISIDpair (50.4.1.4). It contains the VID/I-SID assignments currently advertised by an AAD in LLDP assignment TLVs.

50.4.3.5 aaCurRemAsgnsTlv

This is a structure of type ASGNSstring (50.4.1.1) containing the state machine's current assignment TLV.

50.4.3.6 aaLocAsgnsTlv

This is a structure of type ASGNSstring (50.4.1.1) containing the current LLDP local database assignment TLV (D.2.18) in machine local byte and bit format. This structure is transfered to the LLDP local database object lldpV2Xdot1LocAaAsgnsTlvString in network byte and bit order by the procedure aaSetAsgnsTlv() (50.5.14).

50.4.3.7 aaLocSysTlv

This is a structure of type SYSstring (50.4.1.3) containing the current LLDP local database PBBN Auto Attach System TLV (D.2.17) in machine local byte and bit format. This structure is transfered to the LLDP local database object lldpV2Xdot1LocAaSysTlvString in network byte and bit order by the procedure aaSetSysTlv() (50.5.15).

50.4.3.8 aaOperLocAsgnsTlv

This is a structure of type ASGNSstring (50.4.1.1) containing the state machine's operational assignment TLV.

50.4.3.9 aaOperLocSysTlv

This is a structure of type SYSstring (50.4.1.3) containing the state machine's operational PBBN Auto Attach System TLV.

50.4.3.10 aaOperRemAsgnsTlv

This is a structure of type ASGNSstring (50.4.1.1) containing the state machine's operational assignment TLV.

50.4.3.11 aaOperRemSysTlv

This is a structure of type SYSstring (50.4.1.3) containing the state machine's operational PBBN Auto Attach System TLV.

50.4.3.12 aaPendingCompleted

This variable is an AAB boolean used to indicate when a pending operation has completed. When TRUE a pending operation has completed. The conditions setting this boolean to TRUE are implementation dependent. The AAB state machine uses this boolean to process the completion of operations that are delayed pending completion of requests to management or control functions.

50.4.3.13 aaPortId

This variable is set by the system at startup to the if-ref for this port.

50.4.3.14 aaPortTagging

This variable is set by the system to the tagging options currently configured on the port. This variable can take the values specified for the PBBN Auto Attach System TLV tagging field listed in Table D-18.

50.4.3.15 aaRemAsgnsTlv

This is a structure of type ASGNSstring (50.4.1.1) containing the current LLDP remote database assignment TLV (D.2.18) derived from lldpV2Xdot1RemAaAsgnsTlvString (D.5.6). If no remote entry exists or if more than a single remote entry exists then aaRemSysTlv == NULL.

50.4.3.16 aaRemSysTlv

This is a structure of type SYSstring (50.4.1.3) containing the current LLDP remote database PBBN Auto Attach System TLV (D.2.17) derived from lldpV2Xdot1RemAaSysTlvString (D.5.6) for this port. If no remote entry exists or if more than a single remote entry exists then aaRemSysTlv == NULL.

50.4.3.17 aaResetTimer

The Auto Attach state machines use a timer to allow the AAB and AAD to return to synchronization after a reset condition. This timer operates as a countdown timer (i.e., it expires when its value reaches zero). The aaTimer:

a) Has a resolution of one second.
b) Is started by loading an initial integer value, n, where $0 < n \leq 1200$.
c) Is decremented by one per timer tick, as long as $n > 0$; the interval between timer ticks is the same as the timer resolution.
d) Represents the remaining time in the period.

50.4.3.18 aaStatsAssocAttached

This is a statistic counter providing the number of times an association attached.

50.4.3.19 aaStatsAssocFailed

This is a statistic counter providing the number of times an association failed.

50.4.3.20 aaStatsAssocReset

This is a statistic counter providing the number of times an association was reset.

50.4.3.21 aaStatsAssocStandby

This is a statistic counter providing the number of times an association was a standby.

50.4.3.22 aaStatsVidIsidAsgnsAccepted

This is a statistic counter providing the number of VID/ISID assignments accepted by the AAB.

50.4.3.23 aaStatsVidIsidAsgnsRejected

This is a statistic counter providing the number of VID/ISID assignments rejected by the AAB.

50.4.3.24 aaStatsVidIsidAsgnsRequested

This is a statistic counter providing the number of VID/ISID assignments requested by the AAD.

50.4.3.25 aaStatsVidIsidAsgnsWithdrawn

This is a statistic counter providing the number of VID/ISID assignment requests withdrawn by the AAD.

50.4.3.26 aaSysTlvEnable

This variable enables and disables transmission of PBBN Auto Attach System TLVs on this port. This variable is the LLDP MIB object lldpXdot1AaConfigSysTxEnable (D.5.6) or LLDP YANG object tlvs-tx-org-aa-system-enable.aaSystem (D.6.6.7).

PBBN Auto Attach System TLV transmission is controlled by the Auto Attach state machine. When aaSysTlvEnable is FALSE, LLDP does not include the Auto Attach System TLV in its transmissions. When the aaSysTlvEnable is TRUE, LLDP includes the Auto Attach System TLV in its transmissions.

50.4.3.27 aaTopologyError

This is a boolean indicating a topology error. If the port is attached to a point-to-point single peer-to-peer network then aaTopologyError is set to FALSE, otherwise aaTopologyError is set to TRUE. When this variable is set to TRUE, the state machines immediately exit their current state and enter the INIT state.

50.5 State machine functions

The state machine functions described here are expressed in both words and pseudo-code fragments. Code fragments use C style syntax; however, when implementation specific operations are required that extend outside the scope of this standard, a word description is inserted in the pseudo-code and italicized. These functions are intended to describe the external behavior requirements of the AAP, however are not intended to be an implementation of the protocol.

Many of these functions operate on SYSstring and ASGNSstring structures. The functions operating on these structures directly modify the structure instance within the state machine. Some of these functions also provide return codes containing the status of the operations performed.

50.5.1 aaBuildSysTlv(SYSstring tlv)

This function loads the argument tlv with the current parameters for this port. The following values are set in tlv. The function returns with the modified tlv.

```
tlv->state = NOT_READY;
tlv->sysType = aaSystemType;
tlv->portTagging = aaPortTagging;
tlv->reserved = 0;
tlv->portNetId.sysMAC = aaSystemMAC;
tlv->portNetId.reserved = 0;
tlv->portNetId.portId = aaPortId;
```

No errors are reported by this routine, however in an implementation there can be internal errors found.

50.5.2 aaBuildAsgnsTlv(ASGNSstring tlv, VIDISIDpair vidIsidArray[], ASGNSstring operTlv)

This function loads tlv with the current parameters for this port, with the assignment requests contained in vidIsidArray[], and with the current operational status from operTlv. Any new assignment status is set to PENDING. In an AAD the operTlv is an internal copy of the current Auto Attach LLDP remote assignment TLV from the AAB.

The function begins by determining the number of elements in the argument vidIsidArray[]. This number is designated here by *n*. The function then initializes the header variables of the tlv argument as follows:

```
tlv->numAsngs = n;
```

Next the function loads the asgnsArray with the VID/ISID pairs requested by the AAD. For each element of the vidIsidArray[] designated by *i* aaBuildAsgnsTlv() sets:

```
if ( n > 0 ){
       for ( i=0; i < n; i++ ) {
              tlv->asgnsArray[i].status = PENDING;
              tlv->asgnsArray[i].vid = vidIsidArray[i].vid;
              tlv->asgnsArray[i].isid = vidIsidArray[i].isid;
       };
};
```

Next the function loads the asgnsArray with current status for each old VID/ISID pair using the operTlv.

```
for ( i=0; i < n; i++ ) {
       for ( j=0; j < operTlv->numAsgns; j++ ) {
              if (tlv->asgnsArray[i].vid == operTlv->asgnsArray[j].vid &&
               tlv->asgnsArray[i].isid == operTlv->asgnsArray[j].isid ) {
                     tlv->asgnsArray[i].status = operTlv->asgnsArray[j].status;
                     break;
              };
       };
};
```

No errors are reported by this routine, though in an implementation there can be internal system errors.

50.5.3 aaConfigAabAsgn(unsigned vid, unsigned isid)

This function configures the AAB port as a Customer Edge Port (CEP) or a Customer Network Port (CNP), depending on the type of AAD requesting attachment. If the attachment is to an S-VLAN aware AAD then a CNP is provided and the requested S-VLAN is activated and coupled to the requested backbone service instance (BSI, 50.2). If the attachment is to any other AAD type, a CEP is provided and the requested C-VLAN is activated and coupled to the requested BSI. The function uses the state machine variables aaPortId and aaOperRemSysTlv to identify the AAB port where the VLAN attaches to the AAB and to determine the tagging type to use to interface to the AAD. The settings required to configure the AAB are implementation dependent.

To perform the configuration the AAB needs to configure:

a) The Customer Edge Port, Provider Edge Port, and filtering database are configured in the C-VLAN component providing customer service on this AAB port identified by the aaPortId state machine variable and with tagging parameters specified in the aaOperRemSysTlv for this port (50.2).
b) The Customer Network Port, Provider Instance Port, and filtering database are configured in the I-component.
c) The Customer Backbone Port and filtering database are configured in the B-component.
d) If the desired BSI identified by the I-SID is not already coupled to the AAB, the AAB advertises the I-SID to the network using either Shortest Path Bridging (SPB) or requests to a central management plane (PBB) to form an attachment to the desired BSI.
e) The function returns ACCEPT if the configuration is complete or PENDING if the configuration is in progress. Once the pending action is complete the system sets the state machine variable aaPendingCompleted = TRUE.

50.5.4 aaDisableAabVidIsidAsgns(ASGNSstring tlvNew, ASGNSstring tlvOld)

This function determines which VID to I-SID assignments to disable on this port. The function compares the list of VID to I-SID assignments in tlvOld to those in tlvNew and disables all the assignments found in tlvOld with status ACCEPT but either missing in tlvNew or REJECTED in tlvNew. A pseudo-code sketch of aaDisableAabVidIsidAsgns() is as follows:

```
if ( tlvOld == NULL || tlvOld->numAsgns == 0 ) return; // no running assignments
for ( i=0; i < tlvOld->numAsgns; i++ ) {
       match = FALSE;
       if ( tlvNew != NULL && tlvNew->numAsgns != 0 )
              for ( j=0; j < tlvNew->numAsgns; j++ ) {
                     if ( tlvOld->asgnsArray[i].vid == tlvNew->asgnsArray[j].vid
                     && tlvOld->asgnsArray[i].isid==tlvNew->asgnsArray[j].isid )
                     { match = TRUE; k = j; };
              }; // for each j
       if ( match == FALSE ) { // tlvOld vid/isid not in tlvNew
              if ( tlvOld->asgnsArray[i].status == ACCEPT ) {
                     disable tlvOld->asgnsArray[i];
```

```
            };
            /* the disable operation indicates removing all operating state */
            /* within the AAB for this VID to I-SID mapping */
      };
      if ( match == TRUE // tlvOld vid/isid ACCEPT but tlvNew vid/isid REJECT
            && tlvOld->asgnsArray[i].status == ACCEPT
            && tlvOld->asgnsArray[i].status != tlvNew->asgnsArray[k].status )
                  disable tlvOld.asgnsArray[i];
                  /* the disable operation indicates removing all operating */
                  /* state within the AAB for this VID to I-SID mapping */
}; // for each i
```

In the pseudo-code the italicized "*disable tlvOld->asgnsArray[i]*" indicates execution of an implementation dependent operation to disable the VID to I-SID mapping.

50.5.5 aaDisableLldpAaTlvTx()

This function disables LLDP transmission of the Auto Attach TLVs. Pseudo-code for aaDisableLldpAaTlvTx() is as follows:

```
if ( aaSysTlvEnable == ENABLED ||
        aaAsgnsTlvEnable == ENABLED ) {
                aaAsgnsTlvEnable = DISABLED;
                aaSysTlvEnable = DISABLED;
                SomethingChangedLocal();
}
```

50.5.6 aaDisableVidIsidAsgns()

This function disables all active VID to I-SID assignments on this port. All AAS internal state required to activate the VID to I-SID assignment on this port is deleted.

50.5.7 aaDiscAssocAabTblCheck(unsigned port, SYSstring tlv)

This function is used to search aaOperDiscAssocTable[] for an entry where the Auto Attach System MAC address contained within an aaOperDiscAssocTable.remPortNetId object matches the Auto Attach System MAC address contained within tlv.portNetId or for an existing entry for this AAB port. The function returns ASSOC_ATTACHED if no match was found or if the AAD is already discovered on this AAB port. The function returns ASSOC_INVALID if the AAD is already discovered on a different AAB port or this is a new AAD identified on the AAB port.

A pseudo-code sketch of the function is as follows:

```
for ( each i an index to aaOperDiscAssocTable[] ) {
      if ( aaOperDiscAssocTable[i].remPortNetId == tlv->portNetId &&
            aaOperDiscAssocTable[i].locPortId == port ) return ASSOC_ATTACHED;
      if ( aaOperDiscAssocTable[i].remPortNetId.sysMAC
            == tlv->portNetId.sysMAC ) return ASSOC_INVALID;
      if ( aaOperDiscAssocTable[i].locPortId == port ) return ASSOC_INVALID;
} //for each i
return ASSOC_ATTACHED;
```

Within the pseudo-code, the italicized text for operator "*each i an index to aaOperDiscAssocTable[]*" replaces the normal C code for loop conditions because the structure of the table aaOperDiscAssocTable is implementation dependent. The for loop within the pseudo-code executes for each member of the aaOperDiscAssocTable, regardless of the table implementation.

50.5.8 aaDiscAssocAadTblCheck(unsigned port, SYSstring tlv)

An AAD can only have a single active association. It is assumed that the aaOperDiscAssocTable[] has already inserted a discovery entry for this port. The aaOperDiscAssocTable[] entry for the active association is indexed by the variable aaOperAadActiveAssocIndex. The function returns ASSOC_ATTACHED if there is no other active association or if the current active association is this local port with this remote port. Otherwise the function returns ASSOC_STANDBY.

```
if ( aaOperAadActiveAssocIndex == NO_ACTIVE_ASSOC ) {
       for ( each i an index of aaOperDiscAssocTable[] ) {
              if ( aaOperDiscAssocTable[i].locPortId == port &&
               aaOperDiscAssocTable[i].remPortNetId == tlv->portNetId ) {
                     aaOperAadActiveAssocIndex = i; return ASSOC_ATTACHED;
              };
       };
} else {
       i= aaOperAadActiveAssocIndex;
       if ( aaOperDiscAssocTable[i].locPortId == port &&
        aaOperDiscAssocTable[i].remPortNetId == tlv->portNetId )
              return ASSOC_ATTACHED;
}
return ASSOC_STANDBY;
```

Within the pseudo-code, the italicized text for operator "*each i an index to aaOperDiscAssocTable[]*" replaces the normal C code for loop conditions because the structure of the table aaOperDiscAssocTable is implementation dependent. The for loop within the pseudo-code executes for each member of the aaOperDiscAssocTable, regardless of the table implementation.

50.5.9 aaDiscAssocTblDelete(unsigned port)

This function deletes an entry in aaOperDiscAssocTable[]. The function deletes the table entry *i* matching the condition aaOperDiscAssocTable[i].locPortId == port.

50.5.10 aaDiscAssocTblInsert(unsigned port, SYSstring tlv)

This function inserts a new entry in aaOperDiscAssocTable[]. The function builds a DISCport entry and inserts it into a new table entry *i*. It is assumed the table is already checked to assure a duplicate entry is not present. The entry is as follows:

```
aaOperDiscAssocTable[i].locPortId = port;
aaOperDiscAssocTable[i].remPortNetId = tlv->portNetID;
```

50.5.11 aaEnableAabVidIsidAsgns(ASGNSstring tlv)

This function determines which VID to I-SID assignments to activate. The function is responsible for determining if each pending VID to I-SID pair contained in the tlv argument can be activated. In addition, this function builds all state required by the AAB to activate a VID to I-SID assignment.

The function modifies the status associated with each VID to I-SID pair to reflect the action taken on the pair. The codes that can be returned are listed in Table D-20. A pseudo-code sketch is as follows:

```
if ( tlv == NULL || tlv->numAsgns == 0 ) return;
for ( i=0; i < tlv->numAsgns; i++ ) {
       if ( tlv->asgnsArray[i].status == PENDING ) {
        if ( the VID to I-SID assignment is acceptable to the AAB ) {
              st=aaConfigAabAsgn(tlv->asgnsArray[i].vid,
                                       tlv->asgnsArray[i].isid);
              /* The aaConfigAabAsgn() builds all operating state */
              /* within the AAB for this VID to I-SID mapping */
              tlv->asgnsArray[i].status = st;
        } else tlv->asgnsArray[i].status = REJECT; //including reason
       };
}; //for each i
```

In this pseudo-code the condition "t*he VID to I-SID assignment is acceptable to the AAB*" is a catch-all for any internal checks the AAB performs to allow the assignment before attempting configuration.

50.5.12 aaEnableAadVidIsidAsgns(ASGNSstring tlv)

This function enables VID to I-SID assignments. It sets all of the AAD internal state required to enable or disable each VID to I-SID pair. Each pair marked with tlv->asgnsArray[i].status == ACCEPT is enabled, otherwise the pair is disabled. The operations required to enable or disable the identified VLAN are system dependent and so not explicitly described in the pseudo-code. Instead the italicized *enable* and *disable* operations are used to indicate the implementation dependent operations.

```
if ( tlv == NULL || tlv->numAsgns == 0 ) return; // no assigns received
for ( i=0; i < tlv->numAsgns; i++ ) {
       if ( tlv->asgnsArray[i].status == ACTIVE )
              enable the VLAN identified by tlv.asgnsArray[i].vid
       else   disable the VLAN identified by tlv.asgnsArray[i].vid;
};
```

50.5.13 aaPortClearStats()

This function clears all the statistic counters for this port.

50.5.14 aaSetAsgnsTlv(ASGNSstring tlv)

The function is used to place a state machine internal structure of type ASGNSstring into the LLDP database. The ASGNSstring referenced by tlv is transformed from the local machine byte and bit order to the assignment TLV byte and bit order as it is moved into the LLDP database for this port.

The function aaSetAsgnsTlv() locks out LLDP from building an LLDPDU during the update. Once the TLV is transfered and LLDP unlocked, the routine sets aaAsgnsTlvEnable = ENABLED then calls SomethingChangedLocal() and returns.

50.5.15 aaSetSysTlv(SYSstring tlv)

The aaSetSysTlv() function is used to place a state machine internal structure of type SYSstring into the LLDP database. The object referenced by tlv is transformed from the local machine byte and bit order to the PBBN Auto Attach System TLV byte and bit order as it is moved into the LLDP database for this port.

The function aaSetSysTlv() locks out LLDP from building an LLDPDU during the update. Once the TLV is transfered and LLDP unlocked, the routine sets aaSysTlvEnable = ENABLED then calls SomethingChangedLocal() and returns.

50.5.16 aaSomethingChangedRemote()

This procedure is called by the LLDP routine SomethingChangedRemote() (IEEE Std 802.1AB-2016 clause 9.2.7.9) when something changes in the remote LLDP database. The procedure performs the following operations:

— The remote LLDP database entries are checked to determine if multiple remote databases are present on this port. If multiple databases are present, then the port is not connected to a point-to-point single peer-to-peer topology and so the procedure sets aaTopologyError = TRUE, aaRemSysTlv = NULL, aaRemAsgnsTlv = NULL. Otherwise the routine sets aaTopologyError = FALSE.

— If aaTopologyError == FALSE, then the procedure updates the aaRemSysTlv as follows:

  If a PBBN Auto Attach System TLV is not present in the remote database or if multiple Auto Attach System TLVs are present, then aaRemSysTlv = NULL, otherwise the contents of the Auto Attach System TLV are transformed from the big-endian format used for network transmission into the local machine format and placed in aaRemSysTlv.

— If aaTopologyError == FALSE, then the procedure updates the aaRemAsgnsTlv as follows:

  If a PBBN Auto Attach Assignment TLV is not present in the remote database or if multiple copies are present, then aaRemAsgnsTlv = NULL, otherwise the contents of the PBBN Auto Attach Assignment TLV are transfered from the big-endian format used for network transmission into the local machine format and placed in aaRemAsgnsTlv.

50.5.17 aaSomethingChangeRemoteInit()

This procedure used to initialize the LLDP callback routine aaSomethingChangedRemote(). The function registers aaSomethingChangedRemote() with LLDP for call back by the LLDP routine SomethingChangedRemote() and initializes aaTopologyError = FALSE, then calls aaSomethingChangedRemote() to initialize the variables aaTopologyError, aaRemSysTlv, and aaRemAsgnsTlv.

50.5.18 aaUpdateReqDelVidIsidStats(ASGNSstring tlvNew, ASGNSstring tlvOld)

This routine is used to update the requested and withdrawn VID to I-SID assignment statistics for both the AAB and AAD. The VID to I-SID assignment status indicated in tlvOld is compared with the VID to I-SID status in tlvNew. Only VID to I-SID assignments that are different in tlvOld and tlvNew are considered in the counts. A pseudo-code sketch of aaUpdateReqDelVidIsidStats() is as follows:

```
if ( tlvOld == NULL && tlvNew == NULL ) return;
if ( tlvOld != NULL && tlvOld->numAsgns != 0 )
       if ( tlvNew == NULL || tlvNew->numAsgns == 0 ) {
              aaStatsVidIsidAsgnsWithdrawn =+ tlvOld->numAsgns;
              return;
       };
if ( tlvNew != NULL && tlvNew->numAsgns != 0 )
       if ( tlvOld == NULL || tlvOld->numAsgns == 0 ) {
              aaStatsVidIsidAsgnsRequested =+ tlvNew->numAsgns;
              return;
       };
if ( tlvOld != NULL && tlvOld->numAsgns != 0
&& tlvNew != NULL && tlvNew->numAsgns != 0 ) {
       for ( i=0; i < tlvNew->numAsgns; i++ ) {
              flag = FALSE;
              for ( j=0; j < tlvOld->numAsgns; j++ ) {
               if ( tlvNew->asgnsArray[i].vid == tlvOld->asgnsArray[j].vid
               && tlvNew->asgnsArray[i].isid == tlvOld->asgnsArray[j].isid )
                     flag = TRUE;
              }; // for each j
              if ( flag == FALSE )  aaStatsVidIsidAsgnsRequested++;
       }; // for each i
       for ( i=0; i < tlvOld->numAsgns; i++ ) {
              flag = FALSE;
              for ( j=0; j < tlvNew->numAsgns; j++ ) {
               if ( tlvNew->asgnsArray[i].vid == tlvOld->asgnsArray[j].vid
               && tlvNew->asgnsArray[i].isid == tlvOld->asgnsArray[j].isid )
                     flag = TRUE;
              }; // for each j
              if ( flag == FALSE )  aaStatsVidIsidAsgnsWithdrawn++;
       }; // for each i
};
```

50.5.19 aaUpdateAcptRejVidIsidStats(ASGNSstring tlvNew, ASGNSstring tlvOld)

This routine is used to update the accepted and rejected VID to I-SID assignment statistics for both the AAB and AAD. The VID to I-SID assignment status indicated in tlvOld is compared with the VID to I-SID status in tlvNew. Only VID to I-SID assignments that are different in tlvOld and tlvNew are considered in the counts. A pseudo-code sketch of aaUpdateAcptRejVidIsidStats() is as follows:

```
if ( tlvNew == NULL || tlvNew->numAsgns == 0 ) return;
if ( tlvOld == NULL || tlvOld->numAsgns == 0 ) {
       for ( i=0; i < tlvNew->numAsgns; i++ ) {
              if ( tlvNew->asgnsArray[i].status == ACCEPTED )
                           aaStatsVidIsidAsgnsAccepted++
              else if ( tlvNew->asgnsArray[i].status != PENDING )
                           aaStatsVidIsidAsgnsRejected++;
       };
} else
       for ( i=0; i < tlvNew->numAsgns; i++ ) {
              flag = FALSE;
              for ( j=0; j < tlvOld->numAsgns; j++ ) {
               if ( tlvNew->asgnsArray[i].vid == tlvOld->asgnsArray[j].vid
                     && tlvNew->asgnsArray[i].isid == tlvOld->asgnsArray[j].isid
                     && tlvNew->asgnsArray[i].status ==
                                          tlvOld->asgnsArray[j].status )
                           flag = TRUE;
              }; // for each j
       if ( flag == FALSE ) {
              if ( tlvNew->asgnsArray[i].status == ACCEPTED )
                           aaStatsVidIsidAsgnsAccepted++
              else if ( tlvNew->asgnsArray[i].status != PENDING )
                           aaStatsVidIsidAsgnsRejected++;
       };
}; //for each i
```

50.5.20 SomethingChangedLocal()

This is an LLDP routine that is called when the Auto Attach state machines make updates to the local LLDP database entries. The routine is defined in the IEEE Std 802.1AB state machines.

50.6 AAD state machine

In an implementation that supports auto attachment, the Auto Attach Device (AAD) state machine implements the functions specified by Figure 50-4. The state machine requires the AAD to form a valid association with one AAB. Therefore, all VID/I-SID requests from an AAD are received on one AAB port supporting an independent VID space.

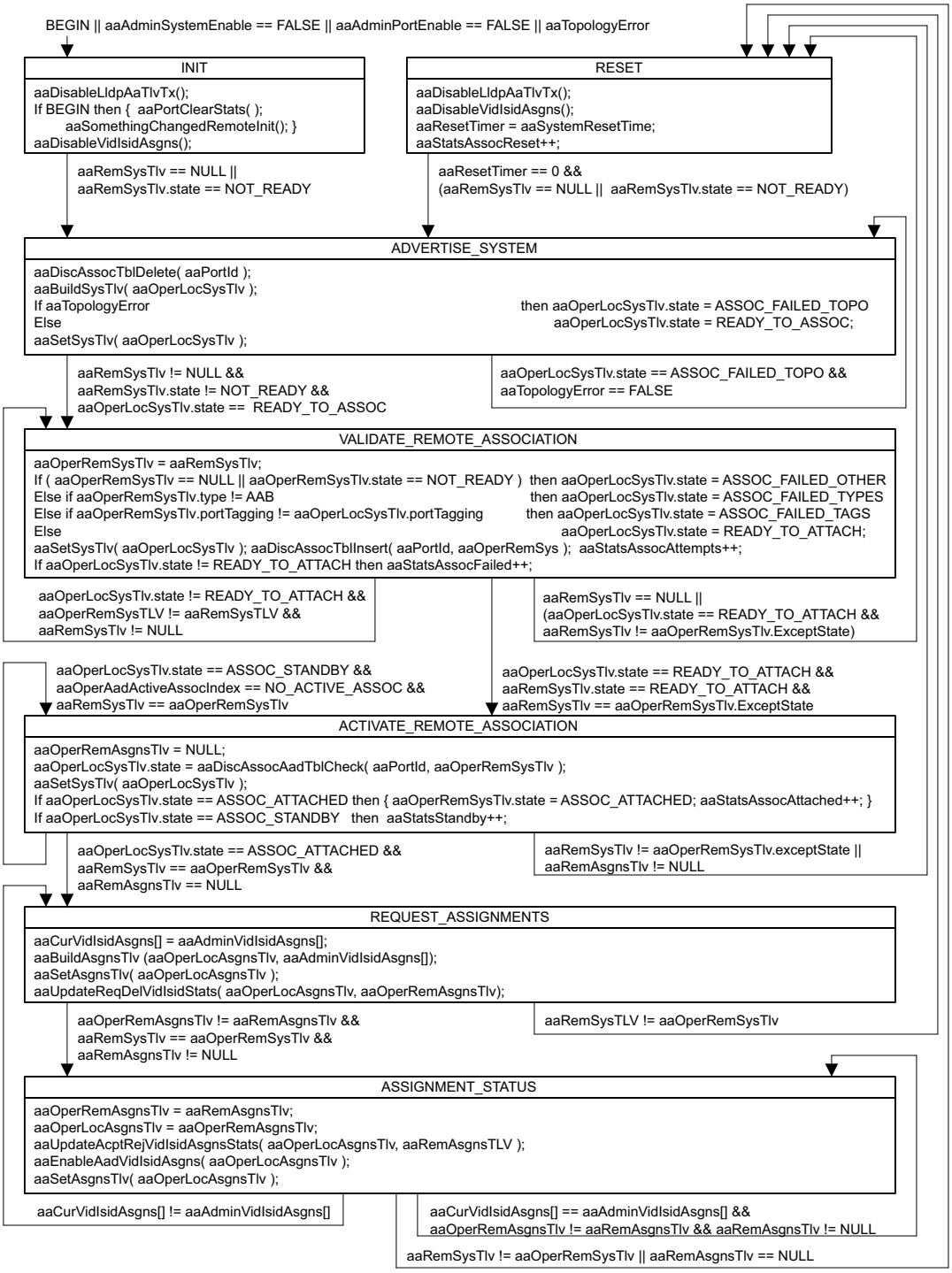

Figure 50-4—PBBN Auto Attach Device (AAD) state machine

50.7 AAB state machine

In an implementation that supports auto attachment, the Auto Attach BEB (AAB) state machine implements the functions specified by Figure 50-5. The state machine requires the AAB to provide a VLAN service specified by the AAD and requires the AAB to have an independent VID space for each port, therefore only one AAD is supported per AAB port.

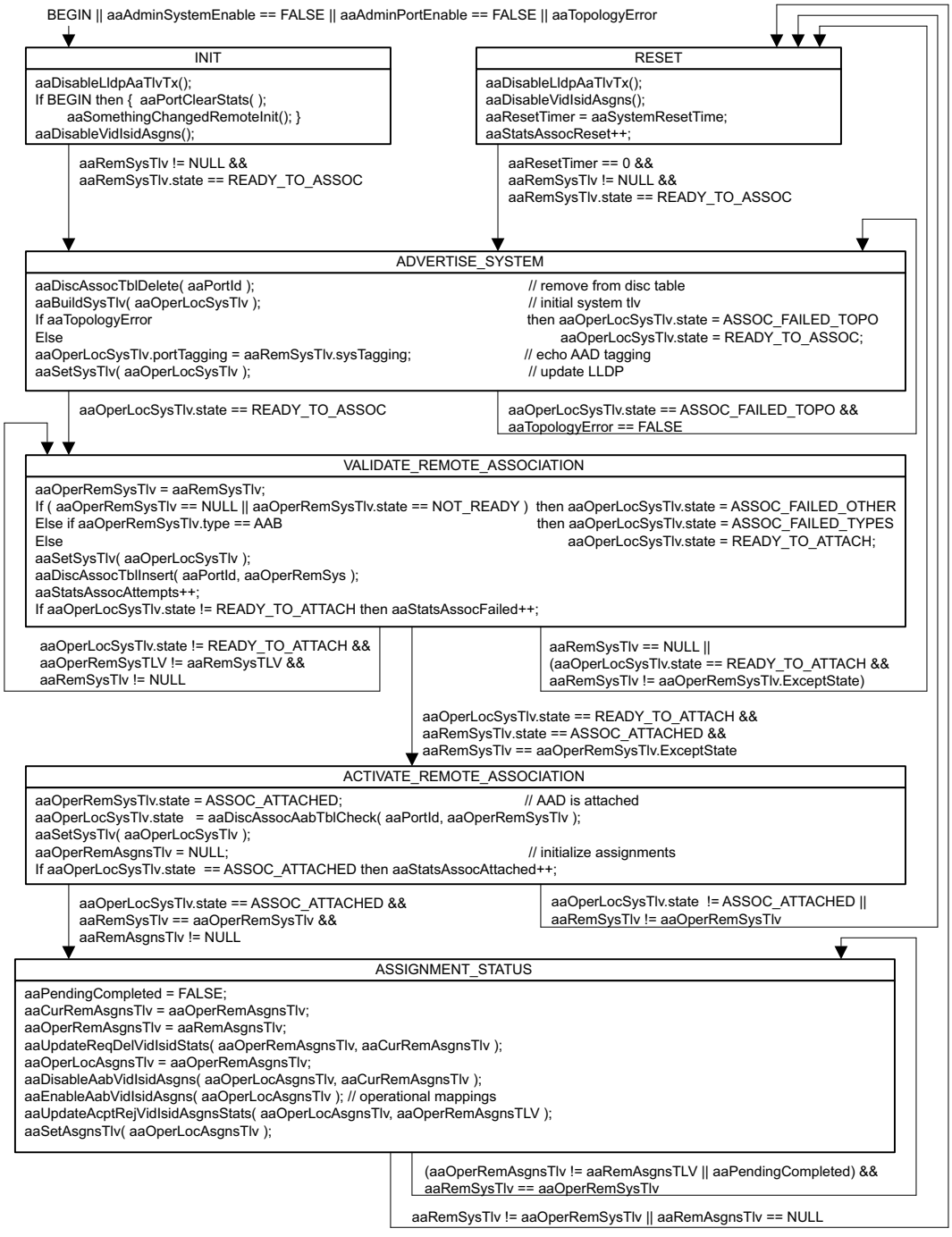

Figure 50-5—PBBN Auto Attach BEB (AAB) state machine

Annex A

(normative)

PICS proforma—Bridge implementations[13]

A.5 Major capabilities

Insert two new rows at the end of the table in A.5 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support
AAB	Does the implementation support AAB functionality?	(BEB-I AND BEB-1):O	Clause 50, 5.12.2	Yes [ ] No [ ] N/A [ ]
AAD	Does the implementation support AAD functionality?	(VLAN OR MBRIDGE):O (BEB OR TMPR):X	Clause 50, 5.9.2, 5.10.3, 5.14.2, 5.33	Yes [ ] No [ ] N/A [ ]

[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.14 Bridge management

Insert a new row at the end of the table in A.14 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support
MGT-253	Does the implementation support the PBBN Auto Attach management entities defined in 12.34?	(AAB OR AAD):M	12.34	Yes [ ] N/A [ ]

A.24 Management Information Base (MIB)

Insert two new rows at the end of the table in A.24 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support
MIB-45	Is the IEEE8021-PBBN-AA-MIB module fully supported (per its MODULE-COMPLIANCE)?	MIB AND (AAB OR AAD):O	17.2.26	Yes [ ] No [ ] N/A [ ]
MIB-46	Are the Auto Attach objects in IEEE8021-LLDP-EXT-DOT1-EVB-EXTENSIONS-MIB module supported (per lldpXdot1AaCompliance MODULE-COMPLIANCE)?	MIB AND (AAB OR AAD):O	D.5.6	Yes [ ] No [ ] N/A [ ]

A.47 YANG

Insert a new row at the end of the table in A.47 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support
YANG-LLDP-PBBN-AA	Is the *ieee802-dot1q-lldp-pbbn-aa-tlv* module supported?	YANG AND (AAB OR AAD): O	D.6.6.7	Yes [ ] No [ ] N/A [ ]

Insert A.54 at end of Annex A, as follows:

A.54 PBBN auto attach

Item	Feature	Status	References	Support
	If auto attach is not supported, mark N/A.	(AAB OR AAD):M	5.9.2, 5.10.3, 5.12.2, 5.14.2, 5.33	N/A [ ]
AAB-1	Does the AAB support one or more I-components?	AAB:M	5.12.2	Yes [ ] N/A [ ]
AAB-2	Does the AAB support the Link Layer Discovery Protocol (LLDP) transmit and receive mode?	AAB:M	IEEE Std 802.1AB™	Yes [ ] N/A [ ]
AAB-3	Does the AAB implementation support the PBBN Auto Attach System TLV?	AAB:M	5.12.2, D.2.17	Yes [ ] N/A [ ]
AAB-4	Does the AAB implementation support the PBBN Auto Attach Assignment TLV?	AAB:M	5.12.2, D.2.18	Yes [ ] N/A [ ]
AAB-5	Does the AAB support the C-tagging interface?	AAB:M	5.12.2, 50.2, D.2.17.8	Yes [ ] N/A [ ]
AAB-6	Does the AAB support the AAB state machine?	AAB:M	5.12.2, 50.7	Yes [ ] N/A [ ]
AAB-7	Does the AAB support the 802.1Q Management VID TLV as specified in D.2.6?	O	D.2.6	Yes [ ] No [ ] N/A [ ]
AAD-1	Does the AAD support the Link Layer Discovery Protocol (LLDP) transmit and receive mode?	AAD:M	IEEE Std 802.1AB™	Yes [ ] N/A [ ]
AAD-2	Does the AAD implementation support the PBBN Auto Attach System TLV?	AAD:M	5.14.2, 5.9.2, D.2.17	Yes [ ] N/A [ ]
AAD-3	Does the AAD implementation support the PBBN Auto Attach Assignment TLV?	AAD:M	5.14.2, 5.9.2, D.2.18	Yes [ ] N/A [ ]
AAD-4	Does the AAD support the AAD state machine?	AAD:M	5.14.2, 5.9.2, 50.6	Yes [ ] N/A [ ]

Annex B

(normative)

PICS proforma—End station implementations[14]

B.5 Major capabilities

Insert a new row at the end of the table in B.5 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support	
AAD	Does the implementation support AAD functionality?	O	5.33, Clause 50	Yes [ ]	No [ ]

Insert new B.20 at the end of Annex B, as follows:

B.20 PBBN Auto Attach

Item	Feature	Status	References	Support
	If PBBN Auto Attach (AAD in B.5) is not supported, mark N/A and ignore the remainder of this table.	AAD:M		N/A[ ]
AAD-1	Does the AAD support the Link Layer Discovery Protocol (LLDP) transmit and receive mode?	AAD:M	IEEE Std 802.1AB™	Yes [ ] N/A [ ]
AAD-2	Does the AAD implementation support the PBBN Auto Attach System TLV?	AAD:M	5.14.2, 5.9.2, D.2.17	Yes [ ] N/A [ ]
AAD-3	Does the AAD implementation support the PBBN Auto Attach Assignment TLV?	AAD:M	5.14.2, 5.9.2, D.2.18	Yes [ ] N/A [ ]
AAD-4	Does the AAD support the AAD state machine?	AAD:M	5.14.2, 5.9.2, 50.6	Yes [ ] N/A [ ]

[14] *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.

Annex D

(normative)

IEEE 802.1 Organizationally Specific TLVs

D.1 Requirements of the IEEE 802.1 Organizationally Specific TLV sets

Change Table D-1 as follows:

Table D-1—IEEE 802.1 Organizationally Specific TLVs

IEEE 802.1 subtype	TLV name	TLV set name	TLV reference	Feature clause reference
0x01	Port VLAN ID	basicSet	D.2.1	6.9
0x02	Port And Protocol VLAN ID	basicSet	D.2.2	6.12
0x03	VLAN Name	basicSet	D.2.3	12.10.2.1.3
0x04	Protocol Identity	basicSet	D.2.4	D.2.4
0x05	VID Usage Digest	basicSet	D.2.5	D.2.5
0x06	Management VID	basicSet	D.2.6	D.2.6
0x07	Link Aggregation TLV	basicSet	IEEE Std 802.1AX	IEEE Std 802.1AX
0x08	Congestion Notification	cnSet	D.2.7	Clause 33
0x09	ETS Configuration TLV	dcbxSet	D.2.8	Clause 38
0x0A	ETS Recommendation TLV	dcbxSet	D.2.9	Clause 38
0x0B	Priority-based Flow Control Configuration TLV	dcbxSet	D.2.10	Clause 38
0x0C	Application Priority TLV	dcbxSet	D.2.11	Clause 38
0x0D	EVB TLV	evbSet	D.2.12	D.2.12
0x0E	CDCP TLV	evbSet	D.2.13	D.2.13
0x10	Application VLAN TLV	dcbxSet	D.2.14	Clause 38
0x11	LRP ECP Discovery TLV	lrpSet	IEEE Std 802.1CS	IEEE Std 802.1CS
0x12	LRP TCP Discovery TLV	lrpSet	IEEE Std 802.1CS	IEEE Std 802.1CS
0x13	Congestion Isolation TLV	ciSet	D.2.15	49.4.4
0x14	Topology Recognition TLV	trSet	D.2.16	49.5
0x15	PBBN Auto Attach System TLV	aaSet	D.2.17	Clause 50
0x16	PBBN Auto Attach Assignment TLV	aaSet	D.2.18	Clause 50

D.2 Organizationally Specific TLV definitions

Insert D.2.17 and D.2.18 at the end of D.2, as follows:

D.2.17 PBBN Auto Attach System TLV

The PBBN Auto Attach System TLV is an optional TLV that allows Bridges and IEEE 802 LAN stations to discover each other and exchange type and configuration information. Refer to Clause 50 for usage of Auto Attach TLVs.

Figure D-17 shows the format of the PBBN Auto Attach System TLV.

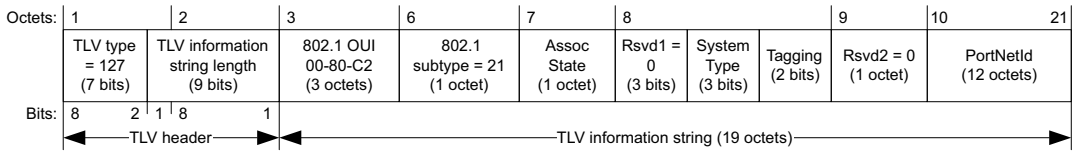

Figure D-17—PBBN Auto Attach System TLV format

D.2.17.1 TLV type

This is a 7-bit integer value occupying the most significant bits of the first octet of the PBBN Auto Attach System TLV. It always contains the value 127.

D.2.17.2 TLV information string length

The TLV information string length field contains the length, in octets, of the fixed length PBBN Auto Attach System TLV fields, which is 19 octets.

D.2.17.3 802.1 OUI

The 3-octet OUI assigned to IEEE 802.1 (00-80-C2).

D.2.17.4 IEEE 802.1 subtype

An octet field containing the PBBN Auto Attach System TLV subtype as specified in Table D-1.

D.2.17.5 Assoc State

An octet field that identifies the current state of the association as specified in Table D-16.

Table D-16—PBBN Auto Attach association states

Assoc State	Value	Notes
NOT_READY	0x00	Not ready to associate
READY_TO_ASSOC	0x01	Ready to associate
READY_TO_ATTACH	0x02	Associated, viable partner
ASSOC_FAILED_TYPES	0x12	Not AAD to AAB
ASSOC_FAILED_TAGS	0x22	Tagging mismatched
ASSOC_FAILED_TOPO	0x32	Multi-point link detected
ASSOC_FAILED_OTHER	0x42	Parsing or resourcing error
ASSOC_ATTACHED	0x03	Association attached
ASSOC_STANDBY	0x13	Association not attached, AAD already attached
ASSOC_INVALID	0x23	Association rejected at AAB because AAD is already attached elsewhere
Reserved	All other	

D.2.17.6 Rsvd1

Three reserved bits, set to zero on transmission. The Rsvd1 field is ignored on receipt.

D.2.17.7 System Type

The System Type field (Table D-17) is a 3-bit unsigned integer used by all AASs to identify the advertising system’s type.

An AAB advertises itself as an AAB system so that connected AADs know that it performs the AAB function of receiving and processing VID/I-SID assignment requests and provides responses to such requests.

An AAD could be a VLAN-aware or VLAN-unaware system. A VLAN-aware system can be a VLAN-aware Bridge, a VLAN-aware wireless LAN access point, or VLAN-aware end station. A VLAN-unaware system can be a MAC Bridge or an end station.

Table D-17—PBBN Auto Attach System Type values

System Type	Value
Auto Attach BEB (AAB)	1
C-VLAN aware Auto Attach Device (AAD)	2
VLAN unaware Auto Attach Device (AAD)	3
S-VLAN aware Auto Attach Device (AAD)	4
Reserved for future standardization	0, 5–7

D.2.17.8 Tagging

The Tagging field (Table D-18) indicates AAD link tagging requirements in AAD-sourced frames and current provisioning mode information. AABs adhere to the Tagging field setting requested by the AAD.

Table D-18—PBBN Auto Attach Tagging field values

Bit Numbers	Name	Value
0–1	Link VLAN Tagging requirements	0 - All traffic tagged on link 1 - Untagged and Tagged traffic on Link 2 - Untagged traffic only on Link 3 - Reserved

D.2.17.9 Rsvd2

One octet reserved for future alternative formats. The octet is set to zero on transmission. All other values are reserved for future standardization. The Rsvd2 field is ignored on receipt.

D.2.17.10 PortNetId

This is a 12-octet field (Table D-19) that uniquely identifies an Auto Attach port within the administrative domain used for Auto Attach connection management. This information is particularly important to an AAD to determine to which AAB in a given network it is attached. AADs only form an association to one AAB, and an AAB forms an association to one AAD per AAB port.

When the AAB function resides on two physical BEBs comprising a DRNI (IEEE Std 802.1AX), the AAD requires an association with a single logical AAB system. AABs in a DRNI configuration use a single unique PortNetId to ensure the AAD is associated logically with one AAB.

Table D-19—PBBN Auto Attach PortNetId values

Octets	Type	Values
6	System MAC Address	
2	Reserved	0 on transmit, can be anything on receive
4	Integer	Port IfIndex, Aggregator IfIndex

D.2.17.11 PBBN Auto Attach System TLV usage rules

An LLDPDU shall contain no more than one PBBN Auto Attach System TLV.

D.2.18 PBBN Auto Attach Assignment TLV

The PBBN Auto Attach Assignment TLV is an optional TLV that allows an AAD to request VID/I-SID assignments that it would like enabled by a directly connected AAB.

Figure D-18 depicts a PBBN Auto Attach Assignment TLV containing multiple VID/I-SID assignments. The minimum number of assignments is zero. The maximum number of assignments is determined by the

maximum TLV information string size of 511 bytes. This allows 101 VID/I-SID assignments resulting in a TLV size of 510 bytes.

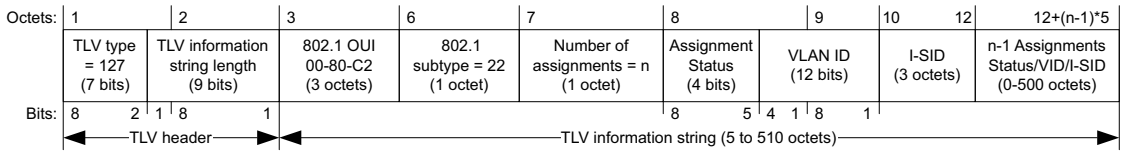

Figure D-18—PBBN Auto Attach Assignment TLV format

D.2.18.1 TLV type

This is a 7-bit integer value occupying the most significant bits of the first octet of the TLV. It always contains the value 127.

D.2.18.2 TLV information string length

The TLV information string length field contains the length, in octets, of the TLV. The length of this TLV is variable between 5 and 510 octets depending on the number of VID/I-SID assignments carried.

D.2.18.3 802.1 OUI

The 3-octet OUI assigned to IEEE 802.1 (00-80-C2).

D.2.18.4 802.1 TLV subtype

An octet containing the PBBN Auto Attach Assignment TLV subtype specified in Table D-1.

D.2.18.5 Number of assignments

An octet containing the number *n* of Status/VID/I-SID triples in this TLV. The number can range from 0 to 101. Zero indicates no Status/VID/I-SID triples are present while 101 is the maximum number allowed in a PBBN Auto Attach Assignment TLV.

D.2.18.6 Assignment Status

The Assignment Status (Table D-20) is returned by the AAB for each VID/I-SID assignment request. If multiple AADs are connected to the AAB, Assignment Status is returned independently on each port. This field is only valid when generated by an AAB.

Table D-20—PBBN Auto Attach Assignment Status values

Assignment Status	Value
Pending	1
Accepted	2
Rejected: Generic	3
Rejected: Auto attach resources unavailable	4
Rejected: Invalid VLAN ID	5
Rejected: VLAN resources unavailable	6
Rejected: Invalid I-SID	7

Table D-20—PBBN Auto Attach Assignment Status values *(continued)*

Assignment Status	Value
Rejected: I-SID resources unavailable	8
Rejected: Application interaction issue	9
Rejected: Assignment not allowed	10

Pending—AAB is processing assignment; used by the AAD while it is waiting for a response from the AAB.

Accepted—AAB assignment request processing is complete for the VID/I-SID and the VLAN to BSI connection has been established.

Rejected: Generic—used when an undefined rejection occurs.

Rejected: Auto attach resources unavailable—used when the Auto Attach process is unable to process a request due to system resources being unavailable. For example, memory or CPU processing.

Rejected: Invalid VLAN ID—used when the VID value is outside the range of 1 to 4094.

Rejected: VLAN resources unavailable—used when VID resource limits have been reached. For example, the maximum number of VLANs are already in use for the port or AAB.

Rejected: Invalid I-SID—used when the I-SID value is outside the allowed values of 1 or 256 through 16 777 214.

Rejected: I-SID resources unavailable—used when I-SID resource limits have been reached. For example, the maximum number of I-SIDs are already in use for the port or AAB.

Rejected: Application interaction issue—used when Auto Attach assignment processing is not able to complete due to an issue with Auto Attach agent functions on the AAB. For example, a state machine issue.

Rejected: Assignment not allowed—used when Auto Attach assignment processing is subject to a policy or rule on the AAB where the assignment requested is not permitted or denied.

D.2.18.7 VLAN ID

This is the VLAN ID of the VLAN being mapped by this attachment. If this is an attachment with a VLAN-unaware AAD, this field is transmitted as zero and ignored on receive. If this is a C-VLAN aware AAD, then this field is a valid C-VID. If this is an S-VLAN aware AAD, then this field is a valid S-VID.

D.2.18.8 I-SID

This is the I-SID for the PBBN backbone service instance (BSI).

D.2.18.9 PBBN Auto Attach Assignment TLV usage rules

An LLDPDU shall contain no more than one PBBN Auto Attach Assignment TLV.

D.3 IEEE 802.1 Organizationally Specific TLV management

Insert D.3.2.13 at the end of D.3, as follows:

D.3.2.13 PBBN Auto Attach TLV managed objects

a) PBBN Auto Attach System TLV (D.2.17).
b) PBBN Auto Attach Assignment TLV (D.2.18).

D.4 PICS proforma for IEEE 802.1 Organizationally Specific TLV extensions[15] [16]

D.4.3 Major capabilities and options

Insert two new rows at the end of the table in D.4.3 as follows (unchanged rows not shown):

Item	Feature	Status	References	Support
dot1aaSet	Is the IEEE 802.1 Organizationally Specific TLV aaSet implemented?	O.1	D.2.17, D.2.18, Table D-1	Yes [ ] No [ ]
dot1aaTlv	Is each TLV in the IEEE 802.1 Organizationally Specific TLV aaSet implemented?	dot1aaSet:M	D.2.17, D.2.18, Table D-1	Yes [ ] No [ ]

[15] Instructions for completing the PICS proforma are given in A.3.

[16] *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.

D.5 IEEE 802.1/LLDP extension MIB

D.5.2 Structure of the IEEE 802.1/LLDP extension MIB

Insert three new rows at the end of Table D-21 (formerly Table D-14, prior to renumbering) as follows (unchanged rows not shown):

Table D-21—IEEE 802.1 extension MIB object group conformance requirements

MIB group	Rx mode	Tx mode	Tx/Rx mode
lldpXdot1CiGroup	CI:M	CI:M	CI:M
lldpXdot1TrGroup	TR:M	TR:M	TR:M
lldpXdot1AaGroup	—	—	AA:M

Insert the following entries at the end of each group in Table D-22 (formerly Table D-15, prior to renumbering). The group headings are shown (in italic) in Table D-15 for reference (other unchanged rows not shown):

Table D-22—IEEE 802.1/LLDP extension MIB object cross reference

MIB table	MIB object	LLDP reference
Configuration group		
lldpXdot1AaConfigAaTable		Augments lldpV2Xdot1LocManVidEntry
	lldpXdot1AaConfigSysTxEnable	Normal LLDPDUs, see IEEE Std 802.1AB
	lldpXdot1AaConfigAsgnsTxEnable	Normal LLDPDUs, see IEEE Std 802.1AB
Local system information		
lldpV2Xdot1LocAaTlvTable		D.2.17, D.2.18
	lldpV2XLocPortIfIndex	(Table index)
	lldpV2Xdot1LocAaSysTlvString	PBBN Auto Attach System TLV string, D.2.17
	lldpV2Xdot1LocAaAsgnsTlvString	PBBN Auto Attach Assignment TLV string, D.2.18

Table D-22—IEEE 802.1/LLDP extension MIB object cross reference *(continued)*

MIB table	MIB object	LLDP reference
Remote system information		
lldpV2Xdot1RemAaTlvTable		D.2.1
	lldpV2RemTimeMark	(Table index)
	lldpV2RemLocalIfIndex	(Table index)
	lldpV2RemLocalDestMACAddress	(Table index)
	lldpV2RemIndex	(Table index)
	lldpV2Xdot1RemAaSysTlvString	PBBN Auto Attach System TLV string, D.2.17
	lldpV2Xdot1RemAaAsgnsTlvString	PBBN Auto Attach Assignment TLV string, D.2.18

D.5.4 Security considerations for IEEE 802.1 LLDP extension MIB module

Change list items g) and h) in D.5.4 as follows:

g) MIB objects that are related to the transmit mode:
   1) lldpV2Xdot1LocPortVlanId
   2) lldpV2Xdot1LocProtoVlanSupported
   3) lldpV2Xdot1LocProtoVlanEnabled
   4) lldpV2Xdot1LocVlanName
   5) lldpV2Xdot1LocProtocolId
   6) lldpV2Xdot1LocVidUsageDigest
   7) lldpV2Xdot1LocManVidTxEnable
   8) lldpV2Xdot1LocLinkAggStatus
   9) lldpV2Xdot1LocLinkAggPortId
   10) lldpXdot1dcbxConfigETSConfigurationEntry
   11) lldpXdot1dcbxConfigPFCTable
   12) lldpXdot1dcbxLocETSBasicConfigurationTable
   13) lldpXdot1dcbxLocETSConPriorityAssignmentTable
   14) lldpXdot1dcbxLocETSRecoTrafficSelectionAlgorithmTable
   15) lldpXdot1dcbxLocPFCBasicTable
   16) lldpXdot1dcbxLocPFCEnableTable
   17) lldpXdot1dcbxAdminETSBasicConfigurationTable
   18) lldpXdot1dcbxAdminETSConPriorityAssignmentTable
   19) lldpXdot1dcbxAdminETSRecoTrafficSelectionAlgorithmTable
   20) lldpXdot1dcbxAdminPFCBasicTable
   21) lldpXdot1dcbxAdminPFCEnableTable
   22) lldpXdot1dcbxLocApplicationPriorityAppTable
   23) lldpXdot1dcbxLocApplicationVlanAppTable
   24) lldpXdot1dcbxAdminApplicationPriorityAppTable
   25) lldpXdot1dcbxAdminApplicationVlanAppTable
   26) lldpXdot1LocCiBasicTable
   27) lldpXdot1CiLocQueueMapTable

28) lldpXdot1LocTrTable
29) lldpV2Xdot1LocAaTlvTable

h) MIB objects that are related to the receive mode:

1) lldpV2Xdot1RemPortVlanId
2) lldpV2Xdot1RemProtoVlanSupported
3) lldpV2Xdot1RemProtoVlanEnabled
4) lldpV2Xdot1RemVlanName
5) lldpV2Xdot1RemProtocolId
6) lldpV2Xdot1RemVidUsageDigest
7) lldpV2Xdot1RemManVidTxEnable
8) lldpV2Xdot1RemLinkAggStatus
9) lldpV2Xdot1RemLinkAggPortId
10) lldpXdot1dcbxConfigETSConfigurationEntry
11) lldpXdot1dcbxConfigPFCTable
12) lldpXdot1dcbxRemETSBasicConfigurationTable
13) lldpXdot1dcbxRemETSConPriorityAssignmentTable
14) lldpXdot1dcbxRemETSRecoTrafficSelectionAlgorithmTable
15) lldpXdot1dcbxRemPFCBasicTable
16) lldpXdot1dcbxRemPFCEnableTable
17) lldpXdot1dcbxAdminETSBasicConfigurationTable
18) lldpXdot1dcbxAdminETSConPriorityAssignmentTable
19) lldpXdot1dcbxAdminETSRecoTrafficSelectionAlgorithmTable
20) lldpXdot1dcbxAdminPFCBasicTable
21) lldpXdot1dcbxAdminPFCEnableTable
22) lldpXdot1dcbxRemApplicationPriorityAppTable
23) lldpXdot1dcbxRemApplicationVlanAppTable
24) lldpXdot1dcbxAdminApplicationPriorityAppTable
25) lldpXdot1dcbxAdminApplicationVlanAppTable
26) lldpXdot1RemCiBasicTable
27) lldpXdot1CiRemQueueMapTable
28) lldpXdot1RemTrTable
29) lldpV2Xdot1RemAaTlvTable

D.5.6 EVB extensions to the IEEE 802.1 LLDP extension MIB module

Change D.5.6 as follows:

In the following MIB definition, should any discrepancy between the DESCRIPTION text and the corresponding definition in D.2.1 through D.5 occur, the definition in D.2.1 through D.5 shall take precedence.

```
LLDP-EXT-DOT1-EVB-EXTENSIONS-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY,
    OBJECT-TYPE
        FROM SNMPv2-SMI
    TruthValue
        FROM SNMPv2-TC
    MODULE-COMPLIANCE,
    OBJECT-GROUP
        FROM SNMPv2-CONF
    ifGeneralInformationGroup
        FROM IF-MIB
    lldpV2LocPortIfIndex,
    lldpV2RemTimeMark,
    lldpV2RemLocalIfIndex,
    lldpV2RemLocalDestMACAddress,
    lldpV2RemIndex,
    lldpV2PortConfigEntry
        FROM LLDP-V2-MIB
    lldpV2Xdot1MIB
        FROM LLDP-EXT-DOT1-V2-MIB;

-- Define the MIB module
        lldpXDot1EvbExtensions MODULE-IDENTITY
    LAST-UPDATED "202211080000Z" -- November 8, 2022"202309210000Z" -- September 21, 2023
    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-chairs@ieee.org"
    DESCRIPTION
            "The LLDP Management Information Base extension module for
            IEEE 802.1 organizationally defined discovery information
            for the EVB extension and PBBN Auto Attach objects.

            This MIB module is rooted under the lldpXdot1StandAloneExtensions
            OID arc, in order to allow it to be defined independently
            of other 802.1 LLDP extension MIBs.

        Unless otherwise indicated, the references in this MIB module are to
        IEEE Std 802.1Q-2022 Bridges and Bridged Networks:
        IEEE Std 802.1Q-2022 as amended by IEEE Std 802.1Qcz, IEEE Std 802.1Qcw,
        and IEEE Std 802.1Qcj.

        Copyright (C) IEEE (20222023).
        This version of this MIB module is part of IEEE Std 802.1Q;
        see that standard for full legal notices."

    REVISION "202309210000Z" -- September 21, 2023
    DESCRIPTION
            "Published as part of IEEE Std 802.1Qcj.
            Auto Attach objects added, references updated."

    REVISION "202211080000Z" -- November 8, 2022
```

```
    DESCRIPTION
            "Published as part of IEEE Std 802.1Q-2022.
            Cross references and contact information updated."

    REVISION "201807010000Z" -- July 1, 2018
    DESCRIPTION
            "Published as part of IEEE Std 802.1Q 2018 revision.
            Cross references updated and corrected."

    REVISION "201412150000Z" -- December 15, 2014
    DESCRIPTION
            "Published as part of IEEE Std 802.1Q 2014 revision.
            Cross references updated and corrected."

    REVISION "201202150000Z" -- February 15, 2012
    DESCRIPTION
            "Initial version published as part of IEEE Std 802.1Qbg"

-- Hang this MIB module under the stand-alone extension MIBs arc:
   ::= { lldpXdot1StandAloneExtensions 1 }

-- Define the root arc for stand-alone extension MIBs in 802.1
lldpXdot1StandAloneExtensions OBJECT IDENTIFIER ::= { lldpV2Xdot1MIB 7 }

-----------------------------------------------------------------------
-----------------------------------------------------------------------
--
-- Organizationally Defined Information Extension - IEEE 802.1
-- Definitions to support the evbSet TLV set (Table D-1)
-- for Edge Virtual Bridging
--
-----------------------------------------------------------------------
-----------------------------------------------------------------------

lldpXdot1EvbMIB OBJECT IDENTIFIER ::= { lldpXDot1EvbExtensions 1 }
lldpXdot1EvbObjects OBJECT IDENTIFIER ::= { lldpXdot1EvbMIB 1 }

-- EVB 802.1 MIB Extension groups

lldpXdot1EvbConfig OBJECT IDENTIFIER ::= { lldpXdot1EvbObjects 1 }
lldpXdot1EvbLocalData OBJECT IDENTIFIER ::= { lldpXdot1EvbObjects 2 }
lldpXdot1EvbRemoteData OBJECT IDENTIFIER ::= { lldpXdot1EvbObjects 3 }

-----------------------------------------------------------------------
-- IEEE 802.1 - EVB Configuration
-----------------------------------------------------------------------
--
--  lldpXdot1EvbConfigEvbTable : configure the
--  transmission of the EVB TLV on a set of ports
--
lldpXdot1EvbConfigEvbTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF LldpXdot1EvbConfigEvbEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A table that controls selection of EVB
         TLVs to be transmitted on individual ports."
    ::= { lldpXdot1EvbConfig 1 }

lldpXdot1EvbConfigEvbEntry OBJECT-TYPE
    SYNTAX        LldpXdot1EvbConfigEvbEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "LLDP configuration information that controls the
        transmission of IEEE 802.1 organizationally defined
        EVB TLV on LLDP transmission-capable ports.

        This configuration object augments the lldpV2PortConfigEntry of
        the LLDP-MIB, therefore it is only present along with the port
        configuration defined by the associated lldpV2PortConfigEntry
        entry.
```

```
        Each active lldpConfigEntry is restored from non-volatile
        storage (along with the corresponding lldpV2PortConfigEntry)
        after a re-initialization of the management system."
    AUGMENTS      { lldpV2PortConfigEntry }
    ::= { lldpXdot1EvbConfigEvbTable 1 }

LldpXdot1EvbConfigEvbEntry ::= SEQUENCE {
    lldpXdot1EvbConfigEvbTxEnable TruthValue
}

lldpXdot1EvbConfigEvbTxEnable OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The lldpXdot1EvbConfigEvbTxEnable, which is
        defined as a truth value and configured by the network
        management, determines whether the IEEE 802.1 organizationally
        defined EVB TLV transmission is allowed
        on a given LLDP transmission-capable port.

        The value of this object is restored from non-volatile
        storage after a re-initialization of the management system."
    REFERENCE
        "D.2.12"
    DEFVAL         { false }
    ::= { lldpXdot1EvbConfigEvbEntry 1 }
--
--  lldpXdot1EvbConfigCdcpTable : configure the
--  transmission of the CDCP TLV on a set of ports
--
lldpXdot1EvbConfigCdcpTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF LldpXdot1EvbConfigCdcpEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A table that controls selection of EVB
         TLVs to be transmitted on individual ports."
    ::= { lldpXdot1EvbConfig 2 }

lldpXdot1EvbConfigCdcpEntry OBJECT-TYPE
    SYNTAX        LldpXdot1EvbConfigCdcpEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "LLDP configuration information that controls the
        transmission of IEEE 802.1 organizationally defined
        CDCP TLV on LLDP transmission-capable ports.

        This configuration object augments the lldpV2PortConfigEntry of
        the LLDP-MIB, therefore it is only present along with the port
        configuration defined by the associated lldpV2PortConfigEntry
        entry.

        Each active lldpConfigEntry is restored from non-volatile
        storage (along with the corresponding lldpV2PortConfigEntry)
        after a re-initialization of the management system."
    AUGMENTS      { lldpV2PortConfigEntry }
    ::= { lldpXdot1EvbConfigCdcpTable 1 }

LldpXdot1EvbConfigCdcpEntry ::= SEQUENCE {
    lldpXdot1EvbConfigCdcpTxEnable TruthValue
}

lldpXdot1EvbConfigCdcpTxEnable OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-write
    STATUS        current
    DESCRIPTION
        "The lldpXdot1EvbConfigCdcpTxEnable, which is
        defined as a truth value and configured by the network
```

```
        management, determines whether the IEEE 802.1 organizationally
        defined CDCP TLV transmission is allowed
        on a given LLDP transmission-capable port.

        The value of this object is restored from non-volatile
        storage after a re-initialization of the management system."
    REFERENCE
        "D.2.13"
    DEFVAL         { false }
    ::= { lldpXdot1EvbConfigCdcpEntry 1 }

--------------------------------------------------------------------------
-- IEEE 802.1 - EVB Local System Information
--------------------------------------------------------------------------
~~~
---
--- lldpV2Xdot1LocEvbTlvTable: EVB TLV Information Table
---
~~~
lldpV2Xdot1LocEvbTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1LocEvbTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of EVB
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the local system
            known to this agent."
    ::= { lldpXdot1EvbLocalData 1 }

lldpV2Xdot1LocEvbTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1LocEvbTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "EVB TLV information about a
            particular port component."
    INDEX   { lldpV2LocPortIfIndex }
    ::= { lldpV2Xdot1LocEvbTlvTable 1 }

LldpV2Xdot1LocEvbTlvEntry ::= SEQUENCE {
      lldpV2Xdot1LocEvbTlvString     OCTET STRING
      }

lldpV2Xdot1LocEvbTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the EVB TLV information string
            for the Port, as defined in ~~D.2.13~~D.2.12.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.12"
    ::= { lldpV2Xdot1LocEvbTlvEntry 1 }
~~~
---
--- lldpV2Xdot1LocCdcpTlvTable: CDCP TLV Information Table
---
~~~
lldpV2Xdot1LocCdcpTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1LocCdcpTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of CDCP
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the local system
```

```
            known to this agent."
    ::= { lldpXdot1EvbLocalData 2 }

lldpV2Xdot1LocCdcpTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1LocCdcpTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "CDCP TLV information about a
            particular port component."
    INDEX   { lldpV2LocPortIfIndex }
    ::= { lldpV2Xdot1LocCdcpTlvTable 1 }

LldpV2Xdot1LocCdcpTlvEntry ::= SEQUENCE {
      lldpV2Xdot1LocCdcpTlvString     OCTET STRING
      }

lldpV2Xdot1LocCdcpTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE(0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the CDCP TLV information string
            for the Port, as defined in ~~D.2.14~~D.2.13.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.13"
    ::= { lldpV2Xdot1LocCdcpTlvEntry 1 }

-- ----------------------------------------------------------------------
-- IEEE 802.1 - EVB Remote System Information
-- ----------------------------------------------------------------------
~~~
---
--- lldpV2Xdot1RemEvbTlvTable: EVB TLV Information Table
---
~~~
lldpV2Xdot1RemEvbTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1RemEvbTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of EVB
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the remote system
            known to this agent."
    ::= { lldpXdot1EvbRemoteData 1 }

lldpV2Xdot1RemEvbTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1RemEvbTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "EVB TLV information about a
            particular port component."
    INDEX   { lldpV2RemTimeMark,
              lldpV2RemLocalIfIndex,
              lldpV2RemLocalDestMACAddress,
              lldpV2RemIndex }
    ::= { lldpV2Xdot1RemEvbTlvTable 1 }

LldpV2Xdot1RemEvbTlvEntry ::= SEQUENCE {
      lldpV2Xdot1RemEvbTlvString     OCTET STRING
      }

lldpV2Xdot1RemEvbTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
```

```
    STATUS      current
    DESCRIPTION
            "This object contains the EVB TLV information string
            for the Port, as defined in ~~D.2.13~~D.2.12.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.12"
    ::= { lldpV2Xdot1RemEvbTlvEntry 1 }
~~~
---
--- lldpV2Xdot1RemCdcpTlvTable: CDCP TLV Information Table
---
~~~
lldpV2Xdot1RemCdcpTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1RemCdcpTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of CDCP
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the remote system
            known to this agent."
    ::= { lldpXdot1EvbRemoteData 2 }

lldpV2Xdot1RemCdcpTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1RemCdcpTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "CDCP TLV information about a
            particular port component."
    INDEX   { lldpV2RemTimeMark,
              lldpV2RemLocalIfIndex,
              lldpV2RemLocalDestMACAddress,
              lldpV2RemIndex }
    ::= { lldpV2Xdot1RemCdcpTlvTable 1 }

LldpV2Xdot1RemCdcpTlvEntry ::= SEQUENCE {
      lldpV2Xdot1RemCdcpTlvString     OCTET STRING
      }

lldpV2Xdot1RemCdcpTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the CDCP TLV information string
            for the Port, as defined in ~~D.2.14~~D.2.13.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.13"
    ::= { lldpV2Xdot1RemCdcpTlvEntry 1 }

-----------------------------------------------------------------------
-- IEEE 802.1 - EVB Conformance Information
-----------------------------------------------------------------------

lldpXdot1EvbConformance OBJECT IDENTIFIER ::= { lldpXDot1EvbExtensions 2 }

lldpXdot1EvbCompliances
    OBJECT IDENTIFIER ::= { lldpXdot1EvbConformance 1 }
lldpXdot1EvbGroups
    OBJECT IDENTIFIER ::= { lldpXdot1EvbConformance 2 }
--
```

```
-- EVB - Compliance Statements
--
lldpXdot1EvbCompliance MODULE-COMPLIANCE
    STATUS        current
    DESCRIPTION
         "A compliance statement for SNMP entities that implement
         the IEEE 802.1 organizationally defined Congestion
         Notification LLDP extension MIB.

         This group is mandatory for agents that implement the
         EVB evbSet TLV set."
    MODULE        -- this module
        MANDATORY-GROUPS  { lldpXdot1EvbGroup,
                            ifGeneralInformationGroup }
    ::= { lldpXdot1EvbCompliances 1 }
--
-- EVB - MIB groupings
--
lldpXdot1EvbGroup  OBJECT-GROUP
    OBJECTS {
      lldpXdot1EvbConfigEvbTxEnable,
      lldpXdot1EvbConfigCdcpTxEnable,
      lldpV2Xdot1LocEvbTlvString,
      lldpV2Xdot1LocCdcpTlvString,
      lldpV2Xdot1RemEvbTlvString,
      lldpV2Xdot1RemCdcpTlvString
      }
    STATUS  current
    DESCRIPTION
        "The collection of objects that support the
        EVB evbSet TLV set."
    ::= { lldpXdot1EvbGroups 1 }

-- ------------------------------------------------------------------------
-- ------------------------------------------------------------------------
--
-- Organizationally Defined Information Extension - IEEE 802.1
-- Definitions to support the aaSet TLV set (Table D-1)
-- for PBBN Auto Attach
--
-- ------------------------------------------------------------------------
-- ------------------------------------------------------------------------

lldpXdot1AaMIB OBJECT IDENTIFIER ::= { lldpXDot1EvbExtensions 3 }
lldpXdot1AaObjects OBJECT IDENTIFIER ::= { lldpXdot1AaMIB 1 }

-- Auto Attach MIB extension groups

lldpXdot1AaConfig OBJECT IDENTIFIER ::= { lldpXdot1AaObjects 1 }
lldpXdot1AaLocalData OBJECT IDENTIFIER ::= {lldpXdot1AaObjects 2 }
lldpXdot1AaRemoteData OBJECT IDENTIFIER ::= { lldpXdot1AaObjects 3 }

--
-- ------------------------------------------------------------------
-- IEEE 802.1 - PBBN Auto Attach configuration
-- ------------------------------------------------------------------
--
--  lldpXdot1AaConfigAaTable : configure the
--  transmission of the PBBN Auto Attach TLVs on a set of ports
--
lldpXdot1AaConfigAaTable OBJECT-TYPE
    SYNTAX        SEQUENCE OF LldpXdot1AaConfigAaEntry
    MAX-ACCESS    not-accessible
    STATUS        current
    DESCRIPTION
        "A table that controls selection of PBBN Auto Attach
         TLVs to be transmitted on individual ports."
    ::= { lldpXdot1AaConfig 1 }

lldpXdot1AaConfigAaEntry OBJECT-TYPE
    SYNTAX        LldpXdot1AaConfigAaEntry
    MAX-ACCESS    not-accessible
    STATUS        current
```

```
    DESCRIPTION
        "LLDP configuration information that controls the
        transmission of IEEE 802.1 organizationally defined
        PBBN Auto Attach System and Assignment TLVs on LLDP
        transmission-capable ports.

        This configuration object augments the lldpV2PortConfigEntry of
        the LLDP-MIB, therefore it is only present along with the port
        configuration defined by the associated lldpV2PortConfigEntry
        entry.

        Each active lldpConfigEntry is restored from non-volatile
        storage (along with the corresponding lldpV2PortConfigEntry)
        after a re-initialization of the management system."
    AUGMENTS      { lldpV2PortConfigEntry }
    ::= { lldpXdot1AaConfigAaTable 1 }

LldpXdot1AaConfigAaEntry ::= SEQUENCE {
    lldpXdot1AaConfigSysTxEnable TruthValue,
    lldpXdot1AaConfigAsgnsTxEnable TruthValue
}

lldpXdot1AaConfigSysTxEnable OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "The lldpXdot1AaConfigSysTxEnable, which is
        defined as a truth value and configured by the PBBN Auto Attach
        state machines, determines whether the IEEE 802.1 organizationally
        defined PBBN Auto Attach System TLV transmission is allowed
        on a given LLDP transmission-capable port.

        The value of this object is returned to FALSE after a re-initialization
        of the management system."
    REFERENCE
        "D.2.17"
    DEFVAL         { false }
    ::= { lldpXdot1AaConfigAaEntry 1 }

lldpXdot1AaConfigAsgnsTxEnable OBJECT-TYPE
    SYNTAX        TruthValue
    MAX-ACCESS    read-only
    STATUS        current
    DESCRIPTION
        "The lldpXdot1AaConfigAsgnsTxEnable, which is
        defined as a truth value and configured by the PBBN Auto Attach
        state machines, determines whether the IEEE 802.1 organizationally
        defined PBBN Auto Attach Assignment TLV transmission is allowed
        on a given LLDP transmission-capable port.

        The value of this object is returned to FALSE after a re-initialization
        of the management system."
    REFERENCE
        "D.2.18"
    DEFVAL         { false }
    ::= { lldpXdot1AaConfigAaEntry 2 }

-- ------------------------------------------------------------------------
-- IEEE 802.1 - PBBN Auto Attach local system information
-- ------------------------------------------------------------------------
---
--- lldpV2Xdot1LocAaTlvTable: PBBN Auto Attach TLVs information table
---
lldpV2Xdot1LocAaTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1LocAaTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of PBBN Auto Attach
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the local system
```

```
            known to this agent."
    ::= { lldpXdot1AaLocalData 1 }

lldpV2Xdot1LocAaTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1LocAaTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "PBBN Auto Attach TLV information about a
            particular port component."
    INDEX   { lldpV2LocPortIfIndex }
    ::= { lldpV2Xdot1LocAaTlvTable 1 }

LldpV2Xdot1LocAaTlvEntry ::= SEQUENCE {
      lldpV2Xdot1LocAaSysTlvString     OCTET STRING,
      lldpV2Xdot1LocAaAsgnsTlvString     OCTET STRING
      }

lldpV2Xdot1LocAaSysTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the PBBN Auto Attach System TLV information
            string for the Port, as defined in D.2.17.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.17"
    ::= { lldpV2Xdot1LocAaTlvEntry 1 }

lldpV2Xdot1LocAaAsgnsTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the PBBN Auto Attach Assignment TLV
            information string for the Port, as defined in D.2.18.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.18"
    ::= { lldpV2Xdot1LocAaTlvEntry 2 }

-- ------------------------------------------------------------------------
-- IEEE 802.1 - PBBN Auto Attach remote system information
-- ------------------------------------------------------------------------
---
--- lldpV2Xdot1RemAaTlvTable: PBBN Auto Attach TLV information table
---
lldpV2Xdot1RemAaTlvTable OBJECT-TYPE
    SYNTAX      SEQUENCE OF LldpV2Xdot1RemAaTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
            "This table contains one row per port of PBBN Auto Attach
            TLV information (as a part of the LLDP
            802.1 organizational extension) on the remote system
            known to this agent."
    ::= { lldpXdot1AaRemoteData 1 }

lldpV2Xdot1RemAaTlvEntry OBJECT-TYPE
    SYNTAX      LldpV2Xdot1RemAaTlvEntry
    MAX-ACCESS  not-accessible
    STATUS      current
    DESCRIPTION
```

```
            "PBBN Auto Attach TLV information about a
            particular port component."
    INDEX   { lldpV2RemTimeMark,
              lldpV2RemLocalIfIndex,
              lldpV2RemLocalDestMACAddress,
              lldpV2RemIndex }
    ::= { lldpV2Xdot1RemAaTlvTable 1 }

LldpV2Xdot1RemAaTlvEntry ::= SEQUENCE {
      lldpV2Xdot1RemAaSysTlvString     OCTET STRING,
      lldpV2Xdot1RemAaAsgnsTlvString   OCTET STRING
      }

lldpV2Xdot1RemAaSysTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the PBBN Auto Attach System TLV
            information string for the Port, as defined in D.2.17.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.17"
    ::= { lldpV2Xdot1RemAaTlvEntry 1 }

lldpV2Xdot1RemAaAsgnsTlvString OBJECT-TYPE
    SYNTAX      OCTET STRING (SIZE (0..514))
    MAX-ACCESS  read-only
    STATUS      current
    DESCRIPTION
            "This object contains the PBBN Auto Attach Assignment TLV
            information string for the Port, as defined in D.2.18.
            As the elements within the string are not individually
            manipulated via SNMP (they are of concern only to the
            state machines), the sub-structure of the string
            is not visible as separate objects within the
            local database."
    REFERENCE
            "D.2.18"
    ::= { lldpV2Xdot1RemAaTlvEntry 2 }

-- ------------------------------------------------------------------------
-- IEEE 802.1 - PBBN Auto Attach conformance information
-- ------------------------------------------------------------------------

lldpXdot1AaConformance OBJECT IDENTIFIER ::= { lldpXDot1EvbExtensions 4 }

lldpXdot1AaCompliances
    OBJECT IDENTIFIER ::= { lldpXdot1AaConformance 1 }
lldpXdot1AaGroups
    OBJECT IDENTIFIER ::= { lldpXdot1AaConformance 2 }
--
-- PBBN Auto Attach - compliance statements
--
lldpXdot1AaCompliance MODULE-COMPLIANCE
    STATUS        current
    DESCRIPTION
         "A compliance statement for SNMP entities that implement
         the IEEE 802.1 organizationally defined PBBN Auto Attach LLDP
         MIB extensions.

         This group is mandatory for agents that implement the
         PBBN Auto Attach aaSet TLV set."
    MODULE        -- this module
        MANDATORY-GROUPS  { lldpXdot1AaGroup,
                            ifGeneralInformationGroup }
    ::= { lldpXdot1AaCompliances 1 }
--
```

```
-- PBBN Auto Attach - MIB groupings
--
lldpXdot1AaGroup  OBJECT-GROUP
    OBJECTS {
      lldpXdot1AaConfigSysTxEnable,
      lldpXdot1AaConfigAsgnsTxEnable,
      lldpV2Xdot1LocAaSysTlvString,
      lldpV2Xdot1LocAaAsgnsTlvString,
      lldpV2Xdot1RemAaSysTlvString,
      lldpV2Xdot1RemAaAsgnsTlvString
      }
    STATUS  current
    DESCRIPTION
        "The collection of objects that support the
        PBBN Auto Attach aaSet TLV set."
    ::= { lldpXdot1AaGroups 1 }

END
```

D.6 IEEE 802.1/LLDP extension YANG

D.6.2 IEEE 802.1 Organizationally Specific TLV YANG data models

Insert D.6.2.7 (including Figure D-26) after D.6.2.6 (inserted by IEEE Std 802.1Qcz-2023) as follows:

D.6.2.7 IEEE 802.1/LLDP extension aaSet TLV model

The attributes for the TLVs in the aaSet are obtained from both system wide and per-port managed objects of the IEEE8021-PBBN-AA-MIB. The model augments the LLDP port model. The UML for the IEEE 802.1/LLDP extension aaSet is derived from the UML specified in IEEE Std 802.1AB and is shown in Figure D-26. The highlighted portions of the UML show how the LLDP model has been augmented.

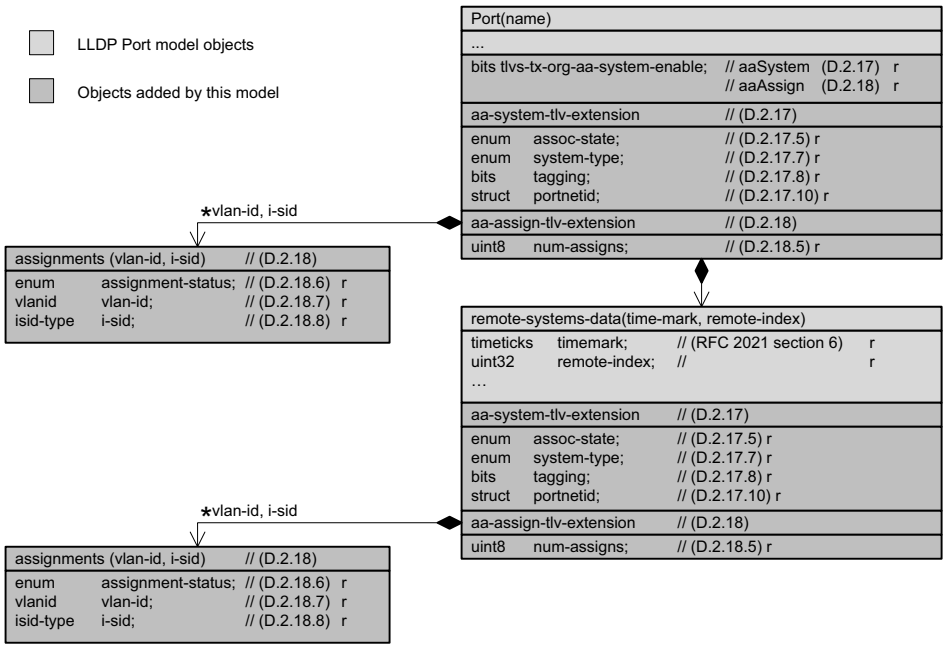

Figure D-26—PBBN Auto Attach aaSet TLV model

D.6.3 Structure of the IEEE 802.1/LLDP extension YANG models

Insert a new row at the end of Table D-23 (inserted by IEEE Std 802.1Qcz-2023 as Table D-16) as follows (unchanged rows not shown):

Table D-23—Summary of the YANG modules

Module	References	Notes
ieee802-dot1q-lldp-pbbn-aa-tlv	D.6.6.7	PBBN Auto Attach set of IEEE 802.1 Organizationally Specific TLVs.

D.6.4 Security considerations

Insert D.6.4.7 after D.6.4.6 (inserted by IEEE Std 802.1Qcz-2023) as follows:

D.6.4.7 Security considerations of the ieee802-dot1q-lldp-pbbn-aa-tlv YANG module

All objects in ieee802-dot1q-lldp-pbbn-aa-tlv are read only and so cannot be manipulated. The objects:

/lldp/port/aa-system-tlv-extension/assoc-state

/lldp/port/aa-system-tlv-extension/system-type

/lldp/port/aa-system-tlv-extension/tagging

/lldp/port/aa-system-tlv-extension/portnetid

/lldp/port/aa-assign-tlv-extension/assignments

provide information about attachments between AADs and AABs that could be useful to an attacker.

D.6.5 Definition of the IEEE 802.1/LLDP extension YANG modules

Insert D.6.5.7 after D.6.5.6 (inserted by IEEE Std 802.1Qcz-2023) as follows:

D.6.5.7 Schema for the ieee802-dot1q-lldp-pbbn-aa-tlv YANG module

```
module: ieee802-dot1q-lldp-pbbn-aa-tlv

  augment /lldp:lldp/lldp:port:
    +--ro tlvs-tx-org-aa-system-enable?   bits
    +--rw aa-system-tlv-extension
    |  +--ro assoc-state?   identityref
    |  +--ro system-type?   identityref
    |  +--ro tagging?       identityref
    |  +--ro portnetid?     port-netid-type
    +--rw aa-assign-tlv-extension
       +--ro num-assigns?   uint8
       +--ro assignments* [vlan-id i-sid]
          +--ro assignment-status?   identityref
          +--ro vlan-id              dot1qtypes:vlanid
          +--ro i-sid                dot1qtypes:isid-type
  augment /lldp:lldp/lldp:port/lldp:remote-systems-data:
    +--ro aa-system-tlv-extension
    |  +--ro assoc-state?   identityref
    |  +--ro system-type?   identityref
    |  +--ro tagging?       identityref
    |  +--ro portnetid?     port-netid-type
    +--ro aa-assign-tlv-extension
       +--ro num-assigns?   uint8
       +--ro assignments* [vlan-id i-sid]
          +--ro assignment-status?   identityref
          +--ro vlan-id              dot1qtypes:vlanid
          +--ro i-sid                dot1qtypes:isid-type
```

D.6.6 IEEE 802.1/LLDP extension YANG modules

Insert D.6.6.7 after D.6.6.6 (inserted by IEEE Std 802.1Qcz-2023) as follows:

D.6.6.7 The ieee802-dot1q-lldp-pbbn-aa-tlv YANG module

```
module ieee802-dot1q-lldp-pbbn-aa-tlv {
  yang-version "1.1";
  namespace urn:ieee:std:802.1Q:yang:ieee802-dot1q-lldp-pbbn-aa-tlv;
  prefix lldp-aa-tlv;
  import ieee802-dot1ab-lldp {
    prefix lldp;
  }
  import ieee802-dot1q-types {
    prefix dot1qtypes;
  }
  organization
    "Institute of Electrical and Electronics Engineers";
  contact
    "WG-URL: http://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-chairs@ieee.org";
  description
    "LLDP PBBN Auto Attach TLVs

    Copyright (C) IEEE (2023).

    This version of this YANG module is part of IEEE Std 802.1Q; see the
    standard itself for full legal notices.";
  revision 2023-10-26 {
    description
      "Published as part of IEEE Std 802.1Qcj-2023.

      The following reference statement identifies each referenced IEEE
      Standard as updated by applicable amendments.";
    reference
      "IEEE Std 802.1Q Bridges and Bridged Networks:
      IEEE Std 802.1Q-2022, IEEE Std 802.1Qcz-2023, IEEE Std 802.1Qcw-2023,
      IEEE Std 802.1Qcj-2023.
      IEEE Std 802 Overview and Architecture: IEEE Std 802-2014.";
  }
  identity assoc-state-selector {
    description
      "Specify the state of the association between the AAS entities as
      described by D.2.17.5 of IEEE Std 802.1Q";
  }
  identity not-ready {
    base assoc-state-selector;
    description
      "Indicates the association state of NOT_READY as specified in D.2.17.5
      of IEEE Std 802.1Q. Signaled as the value 0x00.";
  }
  identity ready-to-assoc {
    base assoc-state-selector;
    description
      "Indicates the association state of READY_TO_ASSOC as specified in
      D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x01.";
  }
  identity ready-to-attach {
    base assoc-state-selector;
    description
      "Indicates the association state of READY_TO_ATTACH as specified in
      D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x02.";
  }
```

```
identity assoc-failed-types {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_FAILED_TYPES as specified
    in D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x12.";
}
identity assoc-failed-tags {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_FAILED_TAGS as specified
    in D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x22.";
}
identity assoc-failed-topo {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_FAILED_TOPO as specified
    in D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x32.";
}
identity assoc-failed-other {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_FAILED_OTHER as specified
    in D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x42.";
}
identity assoc-attached {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_ATTACHED as specified in
    D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x03.";
}
identity assoc-standby {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_STANDBY as specified in
    D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x13.";
}
identity assoc-invalid {
  base assoc-state-selector;
  description
    "Indicates the association state of ASSOC_INVALID as specified in
    D.2.17.5 of IEEE Std 802.1Q. Signaled as the value 0x23.";
}
identity system-type-selector {
  description
    "Specify the selection of the PPBN Auto-attach system type as
    specified in D.2.17.7 of IEEE Std 802.1Q";
}
identity aab-system {
  base system-type-selector;
  description
    "Indicates the system type of an Auto Attach BEB (AAB) in the PPBN
    Auto Attach System TLV specified in D.2.17 of IEEE Std 802.1Q.
    Signaled as the value 1.";
}
identity aad-cvlan-system {
  base system-type-selector;
  description
    "Indicates the system type of a C-VLAN aware Auto Attach Device
    (AAD) in the PPBN Auto Attach System TLV specified in D.2.17 of
    IEEE Std 802.1Q. Signaled as the value 2.";
}
identity aad-vlan-unaware-system {
  base system-type-selector;
  description
    "Indicates the system type of a VLAN unaware Auto Attach Device
    (AAD) in the PPBN Auto Attach System TLV specified in D.2.17 of
    IEEE Std 802.1Q. Signaled as the value 3.";
}
identity aad-svlan-system {
  base system-type-selector;
  description
    "Indicates the system type of a S-VLAN aware Auto Attach Device
```

```
    (AAD) in the PPBN Auto Attach System TLV specified in D.2.17 of
    IEEE Std 802.1Q. Signaled as the value 4.";
}
identity tagging-selector {
  description
    "Specify the tagging field selection of the PPBN Auto Attach System
    TLV as specified in D.2.17.8 of IEEE Std 802.1Q";
}
identity tagged-only {
  base tagging-selector;
  description
    "Indicates the AAD link tagging requirement of all VLAN tagged in
    AAD sourced frames. Signaled as the value 0 in the PPBN Auto Attach
    System TLV specified in D.2.17 of IEEE Std 802.1Q.";
}
identity untagged-or-tagged {
  base tagging-selector;
  description
    "Indicates the AAD link tagging requirement of untagged and VLAN
    tagged in AAD sourced frames. Signaled as the value 1 in the PPBN
    Auto Attach System TLV specified in D.2.17 of IEEE Std 802.1Q.";
}
identity untagged-only {
  base tagging-selector;
  description
    "Indicates the AAD link tagging requirement of all untagged in AAD
    sourced frames. Signaled as the value 2 in the PPBN Auto Attach
    System TLV specified in D.2.17 of IEEE Std 802.1Q.";
}
identity assignment-status-selector {
  description
    "Specify the status of an assignment in the PBBN Auto Attach
    Assignment TLV sent by the AAB for each VID/I-SID assignment request.";
}
identity pending {
  base assignment-status-selector;
  description
    "Indicates the AAB is processing the assignment. This status is
    used by the AAD while it is waiting for a response from the AAB as
    described in D.2.18.6 of IEEE Std 802.1Q. Signaled as the value 1.";
}
identity accepted {
  base assignment-status-selector;
  description
    "Indicates the AAB assignment processing request is complete for
    the VID/I-SID and the VLAN to BSI connection has been established
    as described in D.2.18.6 of IEEE Std 802.1Q. Signaled as the value
    2.";
}
identity rejected-generic {
  base assignment-status-selector;
  description
    "Indicates an undefined rejection has occurred as described in
    D.2.18.6 of IEEE Std 802.1Q. Signaled as the value 3.";
}
identity rejected-resource {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred due to system resources being
    unavailable as described in D.2.18.6 of IEEE Std 802.1Q. Signaled
    as the value 4.";
}
identity rejected-invalid-vlan {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred because the VID value is
    outside the range of 1 to 4094 as described in D.2.18.6 of IEEE Std
    802.1Q. Signaled as the value 5.";
}
identity rejected-vlan-resource {
  base assignment-status-selector;
  description
```

```
    "Indicates a rejection has occurred due to maximum VLAN resource
    limits have been reached as described in D.2.18.6 of IEEE Std
    802.1Q. Signaled as the value 6.";
}
identity rejected-invalid-isid {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred because the I-SID value is
    outside of the range 1 or 256 through 16777214 as described in
    D.2.18.6 of IEEE Std 802.1Q. Signaled as the value 7.";
}
identity rejected-isid-resource {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred due to maximum I-SID resource
    limits have been reached as described in D.2.18.6 of IEEE Std
    802.1Q. Signaled as the value 8.";
}
identity rejected-application {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred because of an issue with Auto
    Attach agent functions on the AAB as described in D.2.18.6 of IEEE
    Std 802.1Q. Signaled as the value 9.";
}
identity rejected-policy {
  base assignment-status-selector;
  description
    "Indicates a rejection has occurred because the Auto Attach
    assignment processing is subject to a policy or rule on the AAB
    where the assignment requested is not permitted or denied as
    described in D.2.18.6 of IEEE Std 802.1Q. Signaled as the value 10.";
}
typedef aa-isid-type {
  type dot1qtypes:isid-type {
    range "1 | 256..16777214";
  }
  description
    "The aa-i-sid type represents a Backbone Service Instance
    Identifier (I-SID). This is the 24-bit I-SID field used in the
    I-TAG TCI of a Provider Backbone Bridging frame. The values 0,
    2-254, and 16777215 are reserved for future standardization. The
    value 255 is dedicated to use by the SPB protocol. For Auto Attach
    values 2-255 are not allowed.";
  reference
    "D.2.18.8 of IEEE Std 802.1Q.";
}
typedef port-netid-type {
  type binary {
    length "12";
  }
  description
    "A 12-octet binary string representing the PortNetID fields of the
    PPBN Auto Attach System TLV as specified in D.2.17 of IEEE Std
    802.1Q. The 12 octet field uniquely identifies an Auto Attach port
    within the administrative domain used for Auto Attach. The first 6
    octets of the string are the binary representation of the system's
    MAC address. The next 2 octets are reserved and have the value 0.
    The last 4 octets are the integer IfIndex for the port.";
  reference
    "D.2.17.10 of IEEE Std 802.1Q";
}
grouping aa-system-tlv {
  description
    "PBBN Auto Attach System TLV";
  reference
    "D.2.17 of IEEE Std 802.1Q";
  leaf assoc-state {
    type identityref {
      base assoc-state-selector;
    }
    config false;
```

```
    description
      "Association state between AAS entities on the link";
    reference
      "D.2.17.5 of IEEE Std 802.1Q.";
  }
  leaf system-type {
    type identityref {
      base system-type-selector;
    }
    config false;
    description
      "Identifies the capability of the advertising system type.";
    reference
      "D.2.17.7 of IEEE Std 802.1Q.";
  }
  leaf tagging {
    type identityref {
      base tagging-selector;
    }
    config false;
    description
      "Indicates AAD link tagging requirements in AAD-sourced frames
      and current provisioning mode information.";
    reference
      "D.2.17.8 of IEEE Std 802.1Q.";
  }
  leaf portnetid {
    type port-netid-type;
    config false;
    description
      "Uniquely identifies an Auto Attach port within the
      administrative domain used for Auto Attach connection management.";
    reference
      "D.2.17.10 of IEEE Std 802.1Q.";
  }
}
grouping aa-assign-tlv {
  description
    "PBBN Auto Attach Assignment TLV";
  reference
    "D.2.18 of IEEE Std 802.1Q";
  leaf num-assigns {
    type uint8 {
      range "0..101";
    }
    config false;
    description
      "Contains the number of Status/VID/I-SID triples in the PBBN Auto
      Attach assignment TLV.";
    reference
      "D.2.18.5 of IEEE Std 802.1Q.";
  }
  list assignments {
    key "vlan-id i-sid";
    config false;
    description
      "A triplet of fields in the PBBN Auto Attach Assignment TLV that
      indicates the mapping of VLAN ID to I-SID and the status of that
      mapping.";
    leaf assignment-status {
      type identityref {
        base assignment-status-selector;
      }
      config false;
      reference
        "D.2.18.6 of IEEE Std 802.1Q.";
    }
    leaf vlan-id {
      type dot1qtypes:vlanid;
      config false;
      description
        "Advertises the VLAN ID of the VLAN being mapped by the
```

```
          assignment. If this is an association with a VLAN unaware AAD
          this field is transmitted as zero and ignored on receive. If
          this is a VLAN aware AAD, then this field is a valid VID.";
        reference
          "D.2.18.7 of IEEE Std 802.1Q.";
      }
      leaf i-sid {
        type aa-isid-type;
        config false;
        description
          "Advertises the value of the PBBN Backbone Service
          Instance Identifier (I-SID) mapped by this assignment.";
        reference
          "D.2.18.8 of IEEE Std 802.1Q.";
      }
    }
  }
  augment "/lldp:lldp/lldp:port" {
    description
      "Augments port with the PBBN Auto Attach TLVs";
    leaf tlvs-tx-org-aa-system-enable {
      type bits {
        bit aaSystem {
          position 0;
          description
            "D.2.17 of IEEE Std 802.1Q";
        }
        bit aaAssign {
          position 1;
          description
            "D.2.18 of IEEE Std 802.1Q";
        }
      }
      config false;
      description
        "Bitmap that includes the aaSet of TLVs from Table D-1 of IEEE
        Std 802.1Q";
      reference
        "D.1 of IEEE Std 802.1Q";
    }
    container aa-system-tlv-extension {
      description
        "The PBBN Auto Attach System TLV";
      uses aa-system-tlv;
    }
    container aa-assign-tlv-extension {
      description
        "The PBBN Auto Attach Assignment TLV";
      uses aa-assign-tlv;
    }
  }
  augment "/lldp:lldp/lldp:port/lldp:remote-systems-data" {
    description
      "Augments port remote-systems-data with received PBBN Auto Attach
      TLVs";
    container aa-system-tlv-extension {
      description
        "Holds a received PBBN Auto Attach System TLV";
      uses aa-system-tlv;
    }
    container aa-assign-tlv-extension {
      description
        "Holds a received PBBN Auto Attach Assignment TLV";
      uses aa-assign-tlv;
    }
  }
}
```

IEEE SA
STANDARDS ASSOCIATION

RAISING THE WORLD'S STANDARDS

Connect with us on:

Twitter: twitter.com/ieeesa
Facebook: facebook.com/ieeesa
LinkedIn: linkedin.com/groups/1791118
Beyond Standards blog: beyondstandards.ieee.org
YouTube: youtube.com/ieeesa

standards.ieee.org
Phone: +1 732 981 0060

IEEE