INTERNATIONAL STANDARD

ISO/IEC/IEEE 8802-1Q

Second edition
2020-08

AMENDMENT 2
2021-09

Telecommunications and exchange between information technology systems — Requirements for local and metropolitan area networks —

Part 1Q: Bridges and bridged networks

AMENDMENT 2: YANG data model

Télécommunications et échange entre systèmes informatiques — Exigences pour les réseaux locaux et métropolitains —

Partie 1Q: Ponts et réseaux pontés

AMENDEMENT 2

Reference number
ISO/IEC/IEEE 8802-1Q:2020/Amd.2:2021(E)

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

Email: stds.ipr@ieee.org
Website: www.ieee.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO/IEC documents should be noted.

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

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

ISO/IEC/IEEE 8802-1Q:2020/Amd 2 was prepared by the LAN/MAN of the IEEE Computer Society (as IEEE Std 802.1Qcp-2018) and drafted in accordance with its editorial rules. It was adopted, under the "fast-track procedure" defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 6, *Telecommunications and information exchange between systems*.

A list of all parts in the ISO/IEC/IEEE 8802 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

IEEE Std 802.1Qcp™-2018
(Amendment to
IEEE Std 802.1Q™-2018)

IEEE Standard for
Local and metropolitan area networks—

Bridges and Bridged Networks—

Amendment 30: YANG Data Model

Sponsor

LAN/MAN Standards Committee
of the
IEEE Computer Society

Approved 14 June 2018

IEEE-SA Standards Board

Abstract: A YANG data model in support of configuration and operational status information for a subset of Bridging capabilities is provided in this amendment to IEEE Std 802.1Q™-2018.

Keywords: Bridged Local Area Networks, IEEE 802®, IEEE 802.1Q™, IEEE 802.1Qcp™, local area networks (LANs), MAC Bridges, metropolitan area networks, Virtual Bridged Local Area Networks (virtual LANs), YANG

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

PDF:	ISBN 978-1-5044-4927-4	STD23138
Print:	ISBN 978-1-5044-4928-1	STDPD23138

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents." They can also be obtained on request from IEEE or viewed at http://standards.ieee.org/IPR/disclaimers.html.

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

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

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. 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.

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

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

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

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

Translations

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

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied 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 his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

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

Comments on standards should be submitted to the following address:

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

Laws and regulations

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

Copyrights

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

Photocopies

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

Updating of IEEE Standards documents

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

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

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

Errata

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

Patents

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

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

Participants

At the time of approval of this standard, the IEEE 802.1 Working Group had the following membership:

Glenn Parsons, *Chair*
John Messenger, *Vice Chair*
János Farkas, *Chair, Time-Sensitive Networking Task Group*
Marc Holness, *Editor*

Ralf Assmann
Shenghua Bao
Jens Bierschenk
Steinar Bjornstad
Christian Boiger
Paul Bottorff
Radhakrishna Canchi
David Chen
Feng Chen
Weiying Cheng
Paul Congdon
Rodney Cummings
Hesham Elbakoury
Norman Finn
Geoffrey Garner
Eric W. Gray
Craig Gunther
Marina Gutierrez
Stephen Haddock
Mark Hantel
Lokesh Kabra
Michael Karl
Stephan Kehrer
Hajime Koto
Christophe Mangin
Scott Mansfield
James McIntosh
Tero Mustala
Tomoki Ohsawa
Donald R. Pannell
Walter Pienciak
Michael Potts
Wei Qiu
Karen Randall
Maximilian Riegel
Jessy V. Rouyer
Atsushi Sato
Frank Schewe
Michael Seaman
Johannes Specht
Patricia Thaler
Paul Unbehagen
Xinyuan Wang
Tongtong Wang
Hao Wang
Karl Weber
Brian Weis
Jordon Woods
Takahiro Yamaura
Xiang Yu
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
Richard Alfvin
Butch Anton
Stefan Aust
Gordon Bechtel
David Black
Christian Boiger
Nancy Bravin
Dietmar Bruckner
Demetrio Bucaneg, Jr.
Stephen Bush
William Byrd
Juan Carreon
Keith Chow
Charles Cook
Rodney Cummings
Sourav Dutta
János Farkas
Norman Finn
Avraham Freedman
Eric W. Gray
Randall Groves
Stephen Haddock
Mark Hantel
Marco Hernandez
Werner Hoelzl
Noriyuki Ikeuchi
Atsushi Ito
Raj Jain
SangKwon Jeong
Peter Jones
Piotr Karocki
Stephan Kehrer
Stuart Kerry
Yongbum Kim
Hyeong Ho Lee
James Lepp
Jon Lewis
Michael Lynch
Elvis Maculuba
Arthur Marris
Richard Mellitz
Charles Moorwood
Nick S. A. Nikjoo
Paul Nikolich
Satoshi Obara
Bansi Patel
Alon Regev
Maximilian Riegel
Robert Robinson
Benjamin Rolfe
Jessy V. Rouyer
Michael Seaman
Daniel Smith
Thomas Starai
Walter Struppler
Mark-Rene Uchida
Dmitri Varsanofiev
George Vlantis
Khurram Waheed
Andreas Wolf
Chun Yu Charles Wong
Oren Yuen
Zhen Zhou

When the IEEE-SA Standards Board approved this standard on 14 June 2018, it had the following membership:

Introduction

This introduction is not part of IEEE Std 802.1Qcp-2018, IEEE Standard for Local and metropolitan area networks—Bridges and Bridged Networks—Amendment 30: YANG Data Model.

This amendment to IEEE Std 802.1Q-2018 specifies a YANG data model that provides configuration and operational status reporting for IEEE 802.1Q Two-Port MAC Relay, Customer VLAN Bridges, and Provider Bridges.

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
USA

Contents

List of figures

List of tables

IEEE Standard for Local and metropolitan area networks—

Bridges and Bridged Networks—

Amendment 30: YANG Data Model

(This amendment is based on IEEE Std 802.1Q™-2018)

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

Editing instructions are shown in ***bold italic***. 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. 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.[1]

1. Overview

1.3 Introduction

Insert the following item after j) Defines SMIv2 (IETF STD 58) Management Information Based (MIB) modules for the management of VLAN Bridge capabilities including spanning tree protocols and Provider Bridges, renumbering as necessary:

k) Define YANG configuration and operational state models (Clause 48) in support of Two-Port MAC Relays, Customer VLAN Bridges, and Provider Bridges.

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

2. Normative references

Insert the following references in alphanumeric order:

IEEE Std 802d™-2017, IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture—Amendment 1: Allocation of Uniform Resource Name (URN) Values in IEEE 802® Standards.

IETF RFC 8343 (proposed standard), A YANG Data Model for Interface Management, March 2018.

4. Abbreviations

Insert the following abbreviations in alphanumeric order in Clause 4:

NETCONF	Network Configuration Protocol
YANG	Yet Another Next Generation[2]

[2] YANG is best viewed as a name, not an acronym.

5. Conformance

5.4 VLAN Bridge component requirements

5.4.1 VLAN Bridge component options

Insert the following item w) after v) Support SMIv2 MIB modules for the management of VLAN Bridge capabilities (Clause 17), renumbering as necessary:

w) Support YANG modules for the management of VLAN Bridge capabilities (Clause 48);

5.13 MAC Bridge component requirements

5.13.1 MAC Bridge component options

Insert the following item h) after g) Support SMIv2 MIB modules for the management of MAC Bridge capabilities (Clause 17), renumbering as necessary:

h) Support YANG modules for the management of MAC Bridge capabilities (Clause 48);

5.16 TPMR conformance

5.16.1 TPMR options

Insert the following item d) after c) Support remote management using the TPMR MIB module defined in 17.7.11, renumbering as necessary:

d) Support remote management using the TPMR YANG module defined in 48.4.2.

8. Principles of Bridge operation

Change the text of 8.12 as follows:

8.12 Bridge Management Entity

~~In order to facilitate interoperable management of Bridges by remote means (as opposed to management via some kind of management console attached directly to the Bridge), it is recommended that SNMP should be the protocol that is used, in conjunction ith the SMIv2 MIB modules specified in Clause 17.~~

Remote management capabilities (as opposed to management via a console or other device attached directly to a Bridge) are modeled as performed by a Bridge Management Entity supported by a protocol stack as described in 8.13.7. To facilitate interoperable remote management, the Bridge Management Entity should use either the SMIv2 MIB modules specified in Clause 17 with SNMP or the YANG modules specified in Clause 48 with a network configuration protocol, e.g., NETCONF, RESTCONF.

8.13 Addressing

Change the text of 8.13.7 as follows:

8.13.7 Bridge Management Entities

~~The recommended protocol for remote Bridge management is SNMP, which typically uses IP as a network layer protocol; however,~~ Network configuration and management protocols typically use IP as a network layer protocol. SNMP can also be supported directly over an IEEE 802 LAN, as specified in IETF RFC 4789. ~~If implemented, the~~ The management protocol stack and address used shall be supported by a single LLC Entity attached to a Bridge Port. The Port should be a Management Port for the Bridge, as described in 8.3 and Figure 8-8, but may be a Port attached to a LAN, as described in 8.13.9, Figure 8-21, and Figure 8-22.

12. Bridge management

12.10 Bridge VLAN managed objects

Change the first paragraph of 12.10.1 as follows:

12.10.1 Bridge VLAN Configuration managed object

The Bridge VLAN Configuration managed object models operations that modify, or inquire about, the overall configuration of the Bridge's VLAN resources. There is a single Bridge VLAN Configuration managed object per Bridge Component.

Insert a new Clause 48 after Clause 47 as follows:

48. YANG Data Model

This clause specifies YANG data models that provide control and status monitoring of IEEE 802.1Q Bridges. Specifically,

a) Two-Port MAC Relays,

b) Customer VLAN bridges, and

c) Provider bridges.

The YANG bridge data management models are derived from UML models specified in 48.3. The UML models are based on Clause 12.

NOTE 1—OMG UML 2.5 [B78] conventions together with C++ language constructs are used in this clause as a representation to convey model structure and relationships.

NOTE 2—The MIB modules specified in Clause 17 were also derived from Clause 12. Consequently, the capabilities and structure of the YANG data models should closely aligned with that represented by the MIBs. However the YANG data model has not been derived from the MIB, and there has been no attempt to include data or modeling constructs that might appear in the MIB but not in the information model.

48.1 YANG Framework

This clause has been developed according to the YANG guidelines published in IETF RFC 6087 [B68] as applicable to IEEE standards.

The YANG framework applies hierarchy in the following areas:

1) The uniform resource name (URN), as specified in IEEE Std 802d. The structure of the URN is such that ieee is the root (i.e., name-space identifier), followed by the standard, then the working group developing the standard.

2) The YANG objects form a hierarchy of configuration and operational data structures that define the YANG model. These hierarchical relationships are described in 48.3.

The general YANG framework hierarchy is illustrated in Figure 48-1.

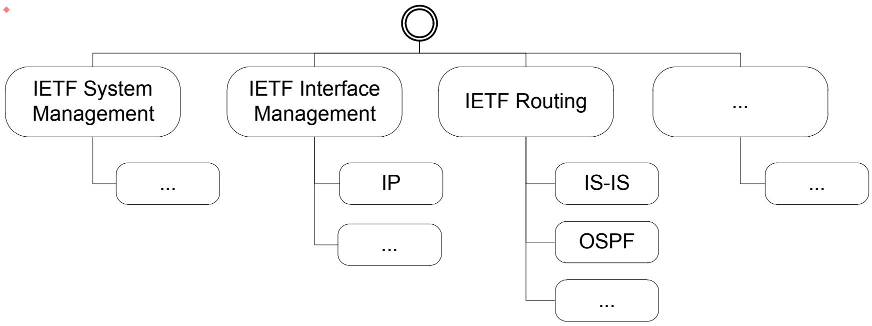


Figure 48-1—General YANG hierarchy

Network interfaces are central to the management of protocols supported over said interface. Thus, it is important to establish a common data model for how interfaces are identified, configured, and monitored. The IETF Interface Management model (IETF RFC 8343) defines a generic YANG data model for the management of network interfaces. Consequently, IEEE 802.1Q Bridge Ports augment the generic interfaces data model defined by the IETF Interface Management model (IETF RFC 8343).

In addition, a high level YANG object is defined to support IEEE 802.1Q Bridge YANG models. An "IEEE 802.1Q Bridge" YANG model is introduced, which can be augmented by the various IEEE 802.1Q Bridge models.

The YANG hierarchical structure that incorporates the IEEE 802.1Q YANG models supported by this standard is represented in Figure 48-2.

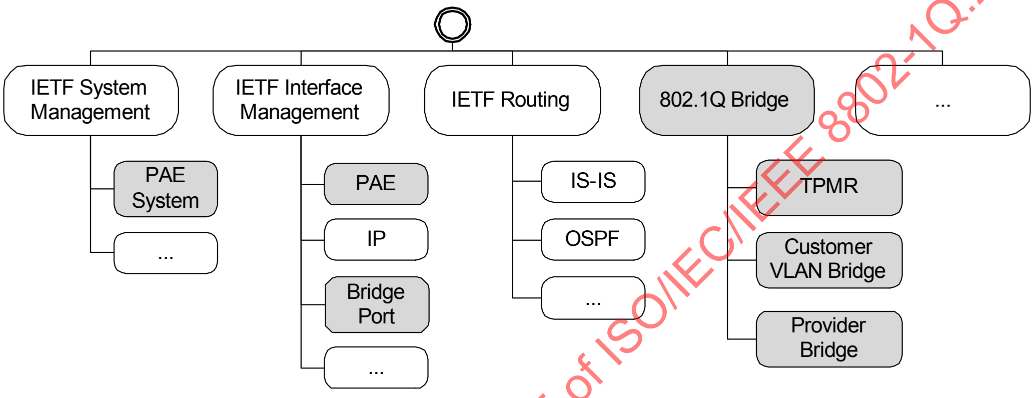


Figure 48-2—YANG root hierarchy with IEEE 802.1Q YANG modules

48.2 Security considerations

The YANG modules defined in this clause are designed to be accessed via a network configuration protocol, e.g., NETCONF protocol (IETF RFC 6536 [B71]). In the case of NETCONF, the lowest NETCONF layer is the secure transport layer and the mandatory to implement secure transport is SSH (IETF RFC 6242 [B70]). The NETCONF access control model provides the means to restrict access for particular NETCONF users to a preconfigured subset of all available NETCONF protocol operations and content.

It is the responsibility of a system's implementor and administrator to ensure that the protocol entities in the system that support NETCONF, and any other remote configuration protocols that make use of these YANG modules, are properly configured to allow access only to those principals (users) that have legitimate rights to read or write data nodes. This standard does not specify how the credentials of those users are to be stored or validated.

48.2.1 Security considerations of the ieee802-dot1q-bridge and ieee802-dot1q-vlan-bridge YANG modules

There are a number of management objects defined in the ieee802-dot1q-bridge and ieee802-dot1q-vlan-bridge YANG modules that are configurable (i.e., read-write) and/or operational (i.e., read-only). Such objects may be considered sensitive or vulnerable in some network environments. A network configuration protocol, such as NETCONF (IETF RFC 6241 [B69]), can support protocol operations that can edit or delete YANG module configuration data (e.g., edit-config, delete-config, copy-config). If this is done in a non-secure environment without proper protection, then negative effects on the network operation is possible.

The following objects in the ieee802-dot1q-bridge and ieee802-dot1q-vlan-bridge YANG modules can be manipulated to interfere with the operation of VLANs and priority classes. This could, for example, be used to force a reinitialization of state machines, thus causing network instability, or to change the forwarding and filtering policies. Another possibility would be for an attacker to override established policy on Port priorities, thus giving a user (or an attacker) unauthorized preferential treatment.

interfaces/interface/bridge-port

interfaces/interface/bridge-port/priority-regeneration

bridges/bridge/component/traffic-class-enabled

bridges/bridge/component/bridge-vlan/protocol-group-database

interfaces/interface/bridge-port/traffic-class

interfaces/interface/bridge-port/protocol-group-vid-set

a) The configurable object bridges/bridge/component/filtering-database/aging-time controls how fast dynamically learned forwarding information is aged out. Setting this object to a large value may simplify FDB overflow attacks. Setting this object to too small a value may compromise the throughput of the network by causing excessive flooding.
b) The configurable object bridges/bridge/component/filtering-database/filtering-entries/entry-type provides a filtering mechanism controlling which Ports frames originating from a specific source may be forwarded to. Write access to this table can be used to turn provisioned filtering off or to add filters to prevent rightful use of the network.

Some of the readable data in this YANG module may be considered sensitive or vulnerable in some network environments. It is thus important to control all types of access (e.g., including NETCONF get, get-config operations) to these objects and possibly to even encrypt the values of these objects when sending them over the network. These tables and objects and their sensitivity/vulnerability are described as follows:

— The object bridges/bridge/component/capabilities could be used by an attacker to determine which attacks might be useful to attempt against a given device.

— The readable objects defined in the ieee802-dot1q-bridge and ieee802-dot1q-vlan-bridge modules provide information about the topology of a bridged network and the attached active stations. The addresses listed in the bridges/bridge/component/filtering-database/filtering-entries usually reveal information about the manufacturer of the MAC hardware, which can be useful information for mounting other specific attacks.

— The readable objects defined in the ieee802-dot1q-bridge and ieee802-dot1q-vlan-bridge modules provide information about the topology of a bridged network and the attached active stations. In some networks, information about attached active stations can be considered personal identifying information about the user of the station. Unauthorized use of this object can be considered a privacy threat.

48.2.2 Security considerations of the ieee802-dot1q-tpmr YANG module

There are a number of management objects defined in the ieee802-dot1q-tpmr YANG module that are configurable (i.e., read-write) and/or operational (i.e., read-only). Such objects may be considered sensitive or vulnerable in some network environments. A network configuration protocol, such as NETCONF (IETF RFC 6241 [B69]), can support protocol operations that can edit or delete YANG module configuration data (e.g., edit-config, delete-config, copy-config). If this is done in a non-secure environment without proper protection, then negative effects on the network operation is possible.

The following objects in the ieee802-dot1q-tpmr YANG module could be manipulated to interfere with the operation of MAC status propagation on a TPMR port and, for example, be used to cause network instability:

interfaces/interface/bridge-port/mac-status-propagation/link-notify
interfaces/interface/bridge-port/mac-status-propagation/link-notify-wait
interfaces/interface/bridge-port/mac-status-propagation/link-notify-retry
interfaces/interface/bridge-port/mac-status-propagation/mac-notify
interfaces/interface/bridge-port/mac-status-propagation/mac-notify-time
interfaces/interface/bridge-port/mac-status-propagation/mac-recover-time

48.2.3 Security considerations of the ieee802-dot1q-pb YANG module

There are a number of management objects defined in the ieee802-dot1q-pb YANG module that are configurable (i.e., read-write) and/or operational (i.e., read-only). Such objects may be considered sensitive or vulnerable in some network environments. A network configuration protocol, such as NETCONF (IETF RFC 6241 [B69]), can support protocol operations that can edit or delete YANG module configuration data (e.g., edit-config, delete-config, copy-config). If this is done in a non-secure environment without proper protection, then negative effects on the network operation is possible.

The following objects in the ieee802-dot1q-pb YANG module can be manipulated to interfere with the operation of VLANs. This could, for example, be used to force a reinitialization of state machines, thus causing network instability, or to change the forwarding and filtering policies.

interfaces/interface/bridge-port
interfaces/interface/bridge-port/cvid-registration
interfaces/interface/bridge-port/service-priority-regeneration

48.3 IEEE 802.1Q YANG model

The YANG data models are based on the management model outline in Clause 12. A UML representation of the management model is provided in the following subclauses.

48.3.1 Interface Management (IETF RFC 8343) Model

The Interface Management YANG model is augmented by the IEEE 802.1Q YANG models, and specifically the Bridge Port. As such, it is useful to have a UML representation. The UML representation of the Interface Management (IETF RFC 8343) model is illustrated in Figure 48-3.[3]

[3] The representation of the Interface YANG model is completely derived from IETF RFC 8343.

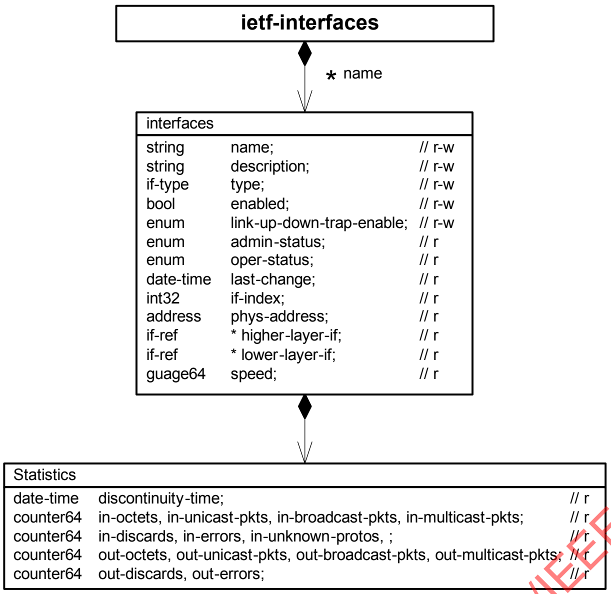

ietf-interfaces
* name
interfaces
string name; // r-w
string description; // r-w
if-type type; // r-w
bool enabled; // r-w
enum link-up-down-trap-enable; // r-w
enum admin-status; // r
enum oper-status; // r
date-time last-change; // r
int32 if-index; // r
address phys-address; // r
if-ref * higher-layer-if; // r
if-ref * lower-layer-if; // r
guage64 speed; // r
Statistics
date-time discontinuity-time; // r
counter64 in-octets, in-unicast-pkts, in-broadcast-pkts, in-multicast-pkts; // r
counter64 in-discards, in-errors, in-unknown-protos, ; // r
counter64 out-octets, out-unicast-pkts, out-broadcast-pkts, out-multicast-pkts; // r
counter64 out-discards, out-errors; // r

Figure 48-3—UML YANG interface management model

48.3.2 Generic IEEE 802.1Q-2018 bridge model

Based upon the bridge management outlined in Clause 12, the generic IEEE 802.1Q bridge model represented in a UML form is illustrated in Figure 48-4.

A Bridge Port represents the service access points to the Bridge's MAC Relay Entity and the interface stack that supports that service access point. Refer to 8.2.

The interface stack associated with the Bridge Port can be such that when it is initially created (a) the underlying MAC entity along with the bridge port data are associated with the same Interface, or (b) the underlying MAC entity is associated with a separate Interface from that of the bridge port Interface. In the later model, the bridge port Interface is the higher layer interface of the underlying MAC entity Interface.

The bridge port interfaces associated with this generic model, is shown in Figure 48-5.

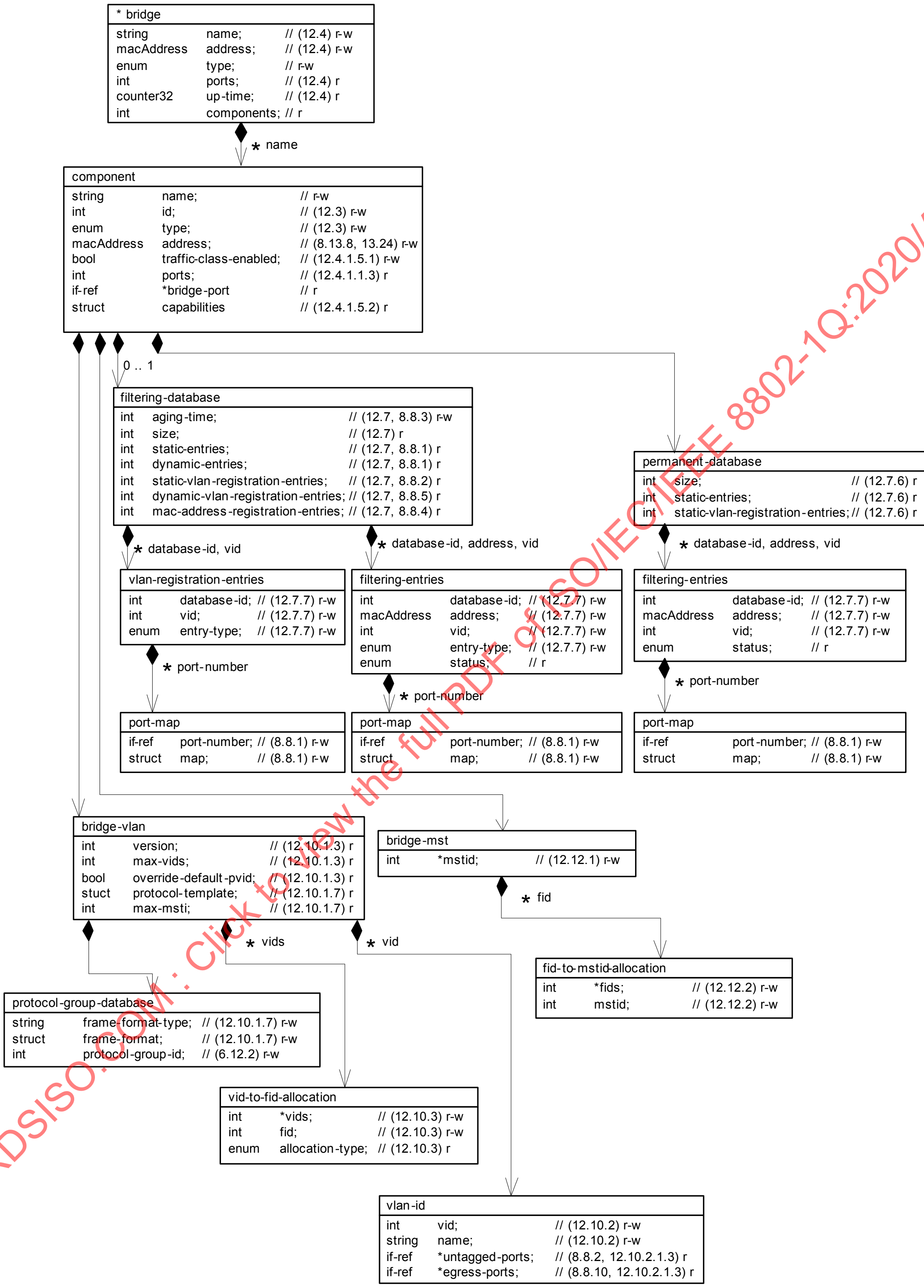
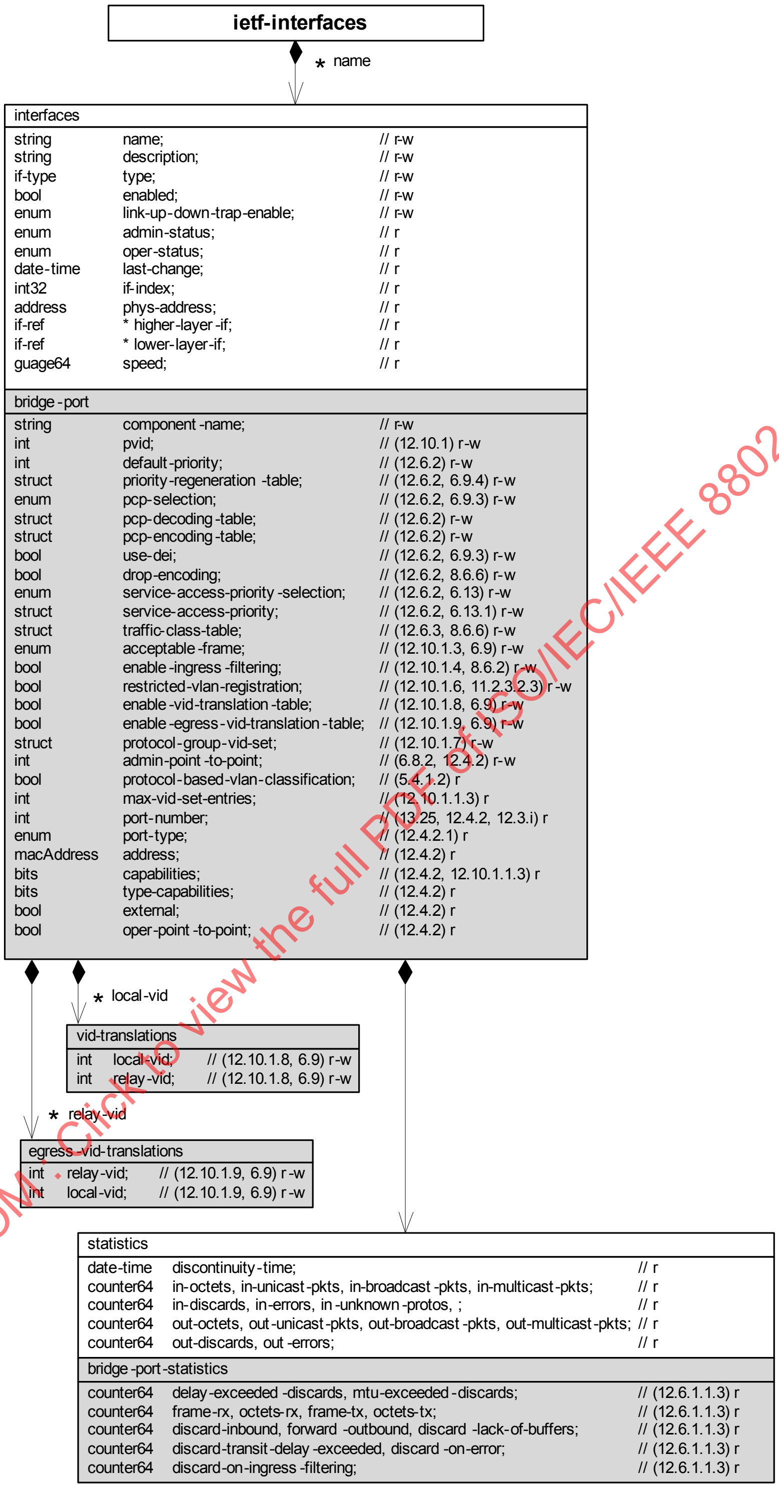

* bridge
string macAddress enum int counter32 int
name; // (12.4) r-w
address; // (12.4) r-w
type; // r-w
ports; // (12.4) r
up-time; // (12.4) r
components; // r
* name
component
string name; // r-w
int id; // (12.3) r-w
enum type; // (12.3) r-w
macAddress address; // (8.13.8, 13.24) r-w
bool traffic-class-enabled; // (12.4.1.5.1) r-w
int ports; // (12.4.1.1.3) r
if-ref *bridge-port // r
struct capabilities // (12.4.1.5.2) r
0 .. 1
filtering-database
int aging-time; // (12.7, 8.8.3) r-w
int size; // (12.7) r
int static-entries; // (12.7, 8.8.1) r
int dynamic-entries; // (12.7, 8.8.1) r
int static-vlan-registration-entries; // (12.7, 8.8.2) r
int dynamic-vlan-registration-entries; // (12.7, 8.8.5) r
int mac-address-registration-entries; // (12.7, 8.8.4) r
permanent-database
int size; // (12.7.6) r
int static-entries; // (12.7.6) r
int static-vlan-registration-entries; // (12.7.6) r
* database-id, vid
* database-id, address, vid
* database-id, address, vid
vlan-registration-entries
int database-id; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum entry-type; // (12.7.7) r-w
filtering-entries
int database-id; // (12.7.7) r-w
macAddress address; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum entry-type; // (12.7.7) r-w
enum status; // r
filtering-entries
int database-id; // (12.7.7) r-w
macAddress address; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum status; // r
* port-number
* port-number
* port-number
port-map
if-ref port-number; // (8.8.1) r-w
struct map; // (8.8.1) r-w
port-map
if-ref port-number; // (8.8.1) r-w
struct map; // (8.8.1) r-w
port-map
if-ref port-number; // (8.8.1) r-w
struct map; // (8.8.1) r-w
bridge-vlan
int version; // (12.10.1.3) r
int max-vids; // (12.10.1.3) r
bool override-default-pvid; // (12.10.1.3) r
stuct protocol-template; // (12.10.1.7) r
int max-msti; // (12.10.1.7) r
bridge-mst
int *mstid; // (12.12.1) r-w
* fid
* vids
* vid
fid-to-mstid-allocation
int *fids; // (12.12.2) r-w
int mstid; // (12.12.2) r-w
protocol-group-database
string frame-format-type; // (12.10.1.7) r-w
struct frame-format; // (12.10.1.7) r-w
int protocol-group-id; // (6.12.2) r-w
vid-to-fid-allocation
int *vids; // (12.10.3) r-w
int fid; // (12.10.3) r-w
enum allocation-type; // (12.10.3) r
vlan-id
int vid; // (12.10.2) r-w
string name; // (12.10.2) r-w
if-ref *untagged-ports; // (8.8.2, 12.10.2.1.3) r
if-ref *egress-ports; // (8.8.10, 12.10.2.1.3) r

Figure 48-4—Generic IEEE 802.1Q bridge mode


ietf-interfaces
* name
interfaces
string name; // r-w
string description; // r-w
if-type type; // r-w
bool enabled; // r-w
enum link-up-down-trap-enable; // r-w
enum admin-status; // r
enum oper-status; // r
date-time last-change; // r
int32 if-index; // r
address phys-address; // r
if-ref * higher-layer-if; // r
if-ref * lower-layer-if; // r
guage64 speed; // r
bridge-port
string component-name; // r-w
int pvid; // (12.10.1) r-w
int default-priority; // (12.6.2) r-w
struct priority-regeneration -table; // (12.6.2, 6.9.4) r-w
enum pcp-selection; // (12.6.2, 6.9.3) r-w
struct pcp-decoding-table; // (12.6.2) r-w
struct pcp-encoding-table; // (12.6.2) r-w
bool use-dei; // (12.6.2, 6.9.3) r-w
bool drop-encoding; // (12.6.2, 8.6.6) r-w
enum service-access-priority-selection; // (12.6.2, 6.13) r-w
struct service-access-priority; // (12.6.2, 6.13.1) r-w
struct traffic-class-table; // (12.6.3, 8.6.6) r-w
enum acceptable-frame; // (12.10.1.3, 6.9) r-w
bool enable-ingress-filtering; // (12.10.1.4, 8.6.2) r-w
bool restricted-vlan-registration; // (12.10.1.6, 11.2.3.2.3) r-w
bool enable-vid-translation-table; // (12.10.1.8, 6.9) r-w
bool enable-egress-vid-translation-table; // (12.10.1.9, 6.9) r-w
struct protocol-group-vid-set; // (12.10.1.7) r-w
int admin-point-to-point; // (6.8.2, 12.4.2) r-w
bool protocol-based-vlan-classification; // (5.4.1.2) r
int max-vid-set-entries; // (12.10.1.1.3) r
int port-number; // (13.25, 12.4.2, 12.3.i) r
enum port-type; // (12.4.2.1) r
macAddress address; // (12.4.2) r
bits capabilities; // (12.4.2, 12.10.1.1.3) r
bits type-capabilities; // (12.4.2) r
bool external; // (12.4.2) r
bool oper-point-to-point; // (12.4.2) r
* local-vid
vid-translations
int local-vid; // (12.10.1.8, 6.9) r-w
int relay-vid; // (12.10.1.8, 6.9) r-w
* relay-vid
egress-vid-translations
int relay-vid; // (12.10.1.9, 6.9) r-w
int local-vid; // (12.10.1.9, 6.9) r-w
statistics
date-time discontinuity-time; // r
counter64 in-octets, in-unicast-pkts, in-broadcast-pkts, in-multicast-pkts; // r
counter64 in-discards, in-errors, in-unknown-protos, ; // r
counter64 out-octets, out-unicast-pkts, out-broadcast-pkts, out-multicast-pkts; // r
counter64 out-discards, out-errors; // r
bridge-port-statistics
counter64 delay-exceeded-discards, mtu-exceeded-discards; // (12.6.1.1.3) r
counter64 frame-rx, octets-rx, frame-tx, octets-tx; // (12.6.1.1.3) r
counter64 discard-inbound, forward-outbound, discard-lack-of-buffers; // (12.6.1.1.3) r
counter64 discard-transit-delay-exceeded, discard-on-error; // (12.6.1.1.3) r
counter64 discard-on-ingress-filtering; // (12.6.1.1.3) r

NOTE—Items shaded in gray indicate extension provided by the IEEE 802.1Q YANG models.

Figure 48-5—Bridge port model

Each Bridge Port (3.25) is associated with one Interface. The *higher-layer-if* and *lower-layer-if* attributes provide the interface stacking relationships of the Bridge Port. For example, an Aggregator (as defined by IEEE Std 802.1AX) Interface could be a Bridge Port. The *lower-layer-if* attributes associated with the Aggregator would point to the Aggregation Ports, thus forming the interface staking relationship of the Bridge Port.

Each Bridge Port is uniquely identified by a Port Number (*port-number*). A port number has no mandatory relationship to an Interface index (if-index).

48.3.2.1 Two-Port MAC Relay model

The Two-Port MAC Relay (TPMR) bridge UML data model is a subset of the generic IEEE 802.1Q bridge illustrated in Figure 48-4. A TPMR:

- Supports only 2 bridge ports.
- Supports a “VLAN-unaware” bridge component.
- Does not support any “bridged VLAN” object.
- Does not support a “filtering database” object.
- Provides extensions to the “bridge port.”

The TPMR model is illustrated in Figure 48-6.

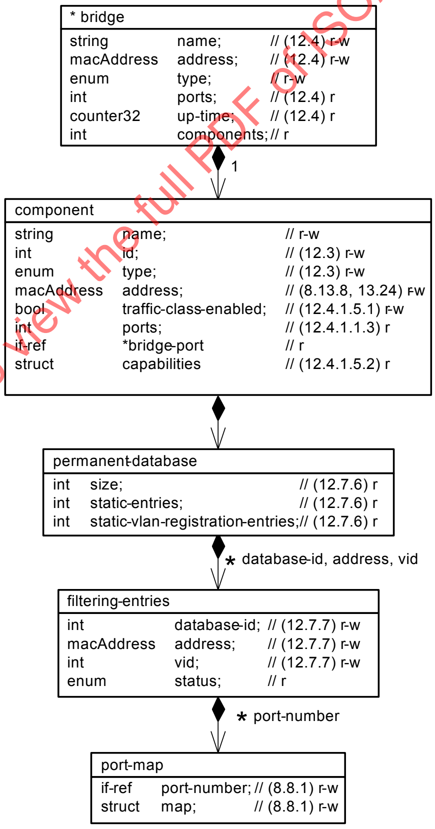
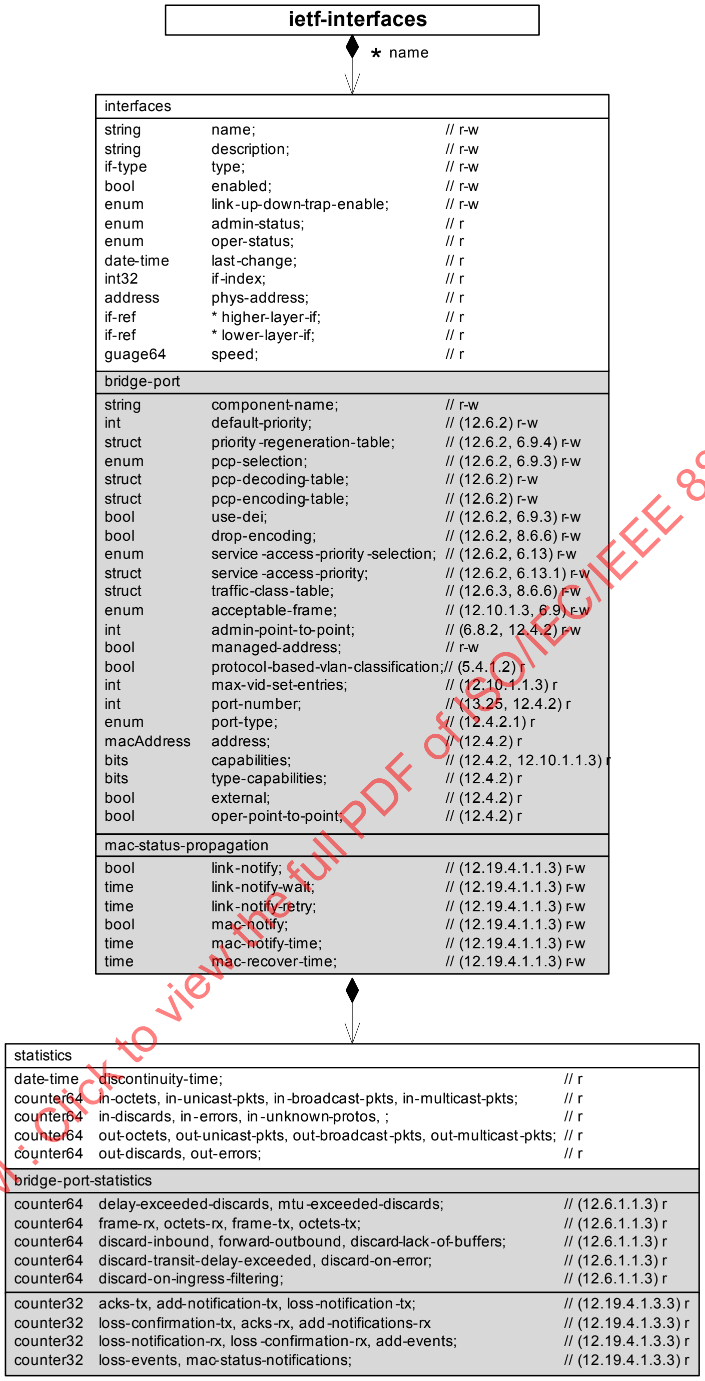

* bridge
string name; // (12.4) r-w
macAddress address; // (12.4) r-w
enum type; // r-w
int ports; // (12.4) r
counter32 up-time; // (12.4) r
int components; // r
1
component
string name; // r-w
int id; // (12.3) r-w
enum type; // (12.3) r-w
macAddress address; // (8.13.8, 13.24) r-w
bool traffic-class-enabled; // (12.4.1.5.1) r-w
int ports; // (12.4.1.1.3) r
if-ref *bridge-port // r
struct capabilities // (12.4.1.5.2) r
permanent-database
int size; // (12.7.6) r
int static-entries; // (12.7.6) r
int static-vlan-registration-entries; // (12.7.6) r
* database-id, address, vid
filtering-entries
int database-id; // (12.7.7) r-w
macAddress address; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum status; // r
* port-number
port-map
if-ref port-number; // (8.8.1) r-w
struct map; // (8.8.1) r-w

Figure 48-6—TPMR model

NOTE—The bridge type would be *two-port-mac-relay-bridge* and the component type would be *d-bridge-component* (which denotes a VLAN unaware component). The number of ports associated with the component must be 2.

The TPMR port UML model is illustrated in Figure 48-7.



interfaces		
string	name;	// r-w
string	description;	// r-w
if-type	type;	// r-w
bool	enabled;	// r-w
enum	link-up-down-trap-enable;	// r-w
enum	admin-status;	// r
enum	oper-status;	// r
date-time	last-change;	// r
int32	if-index;	// r
address	phys-address;	// r
if-ref	* higher-layer-if;	// r
if-ref	* lower-layer-if;	// r
guage64	speed;	// r

bridge-port		
string	component-name;	// r-w
int	default-priority;	// (12.6.2) r-w
struct	priority-regeneration-table;	// (12.6.2, 6.9.4) r-w
enum	pcp-selection;	// (12.6.2, 6.9.3) r-w
struct	pcp-decoding-table;	// (12.6.2) r-w
struct	pcp-encoding-table;	// (12.6.2) r-w
bool	use-dei;	// (12.6.2, 6.9.3) r-w
bool	drop-encoding;	// (12.6.2, 8.6.6) r-w
enum	service-access-priority-selection;	// (12.6.2, 6.13) r-w
struct	service-access-priority;	// (12.6.2, 6.13.1) r-w
struct	traffic-class-table;	// (12.6.3, 8.6.6) r-w
enum	acceptable-frame;	// (12.10.1.3, 6.9) r-w
int	admin-point-to-point;	// (6.8.2, 12.4.2) r-w
bool	managed-address;	// r-w
bool	protocol-based-vlan-classification;	// (5.4.1.2) r
int	max-vid-set-entries;	// (12.10.1.1.3) r
int	port-number;	// (13.25, 12.4.2) r
enum	port-type;	// (12.4.2.1) r
macAddress	address;	// (12.4.2) r
bits	capabilities;	// (12.4.2, 12.10.1.1.3) r
bits	type-capabilities;	// (12.4.2) r
bool	external;	// (12.4.2) r
bool	oper-point-to-point;	// (12.4.2) r

mac-status-propagation		
bool	link-notify;	// (12.19.4.1.1.3) r-w
time	link-notify-wait;	// (12.19.4.1.1.3) r-w
time	link-notify-retry;	// (12.19.4.1.1.3) r-w
bool	mac-notify;	// (12.19.4.1.1.3) r-w
time	mac-notify-time;	// (12.19.4.1.1.3) r-w
time	mac-recover-time;	// (12.19.4.1.1.3) r-w

statistics		
date-time	discontinuity-time;	// r
counter64	in-octets, in-unicast-pkts, in-broadcast-pkts, in-multicast-pkts;	// r
counter64	in-discards, in-errors, in-unknown-protos, ;	// r
counter64	out-octets, out-unicast-pkts, out-broadcast-pkts, out-multicast-pkts;	// r
counter64	out-discards, out-errors;	// r

bridge-port-statistics		
counter64	delay-exceeded-discards, mtu-exceeded-discards;	// (12.6.1.1.3) r
counter64	frame-rx, octets-rx, frame-tx, octets-tx;	// (12.6.1.1.3) r
counter64	discard-inbound, forward-outbound, discard-lack-of-buffers;	// (12.6.1.1.3) r
counter64	discard-transit-delay-exceeded, discard-on-error;	// (12.6.1.1.3) r
counter64	discard-on-ingress-filtering;	// (12.6.1.1.3) r
counter32	acks-tx, add-notification-tx, loss-notification-tx;	// (12.19.4.1.3.3) r
counter32	loss-confirmation-tx, acks-rx, add-notifications-rx	// (12.19.4.1.3.3) r
counter32	loss-notification-rx, loss-confirmation-rx, add-events;	// (12.19.4.1.3.3) r
counter32	loss-events, mac-status-notifications;	// (12.19.4.1.3.3) r

NOTE—Items shaded in gray indicate extensions provided by the IEEE 802.1Q YANG models.

Figure 48-7—TPMR port model

48.3.2.2 Customer VLAN Bridge model

The Customer VLAN Bridge UML data model is a subset of the generic IEEE 802.1Q bridge data model illustrated in Figure 48-4. The bridge ports UML model associated with a Customer VLAN Bridge is also represented by the generic bridge port model illustrated in Figure 48-5.

A Customer VLAN Bridge:

- Supports a single bridge component.
- Supports a "C-VLAN" bridge component.

48.3.2.3 Provider Bridge model

The Provider Bridge UML data model leverages the generic IEEE 802.1Q bridge illustrated in Figure 48-4. A Provider Bridge:

- Supports a multiple bridge component.
- Specifically, multiple C-VLAN components and one S-VLAN component are used.
- An intra-LAN interface is required to connect the C-VLAN component(s) and S-VLAN component.
- Specific extensions are needed to the "bridge port," associated with each bridge component.

The Provider Bridge model is illustrated in Figure 48-8.

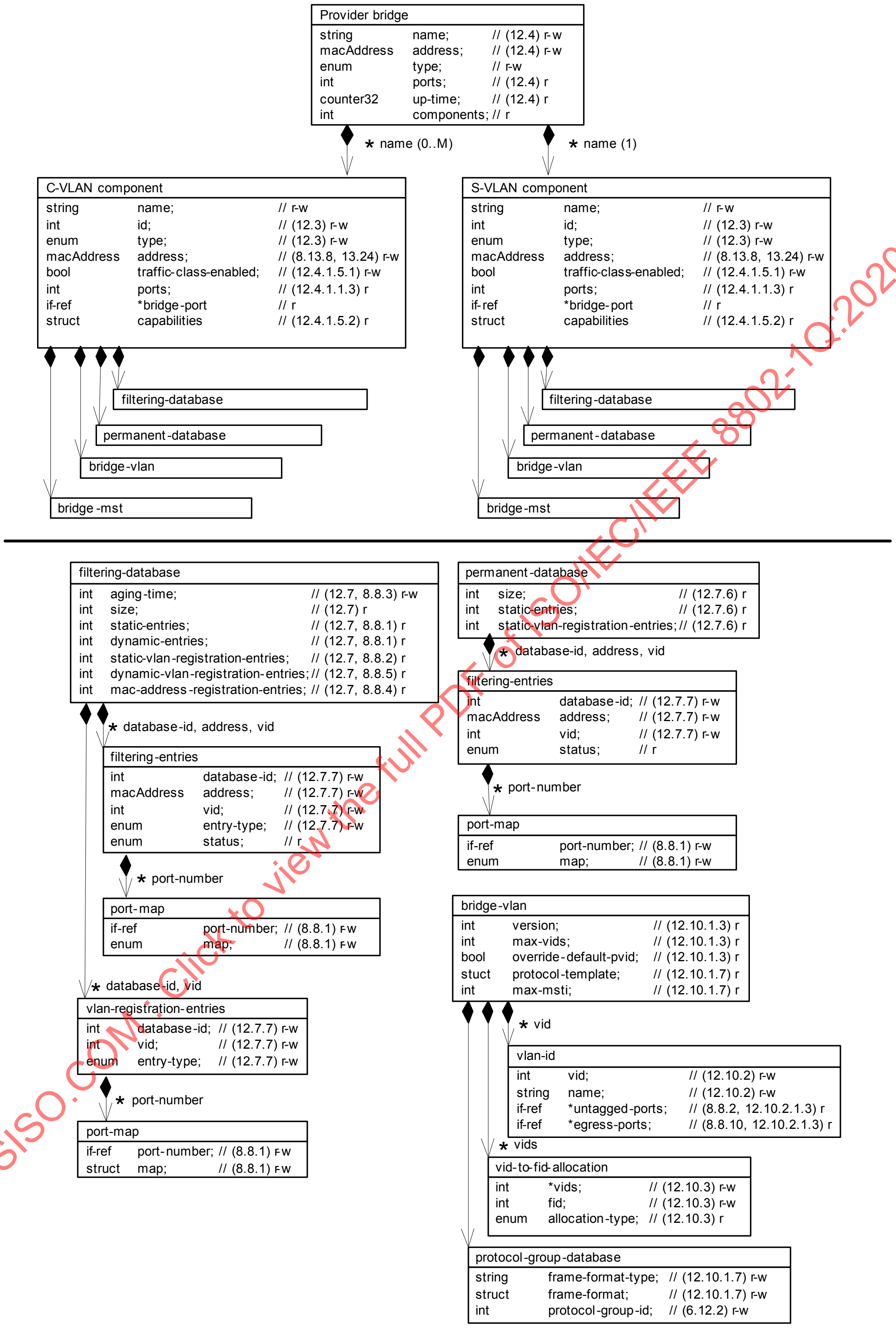
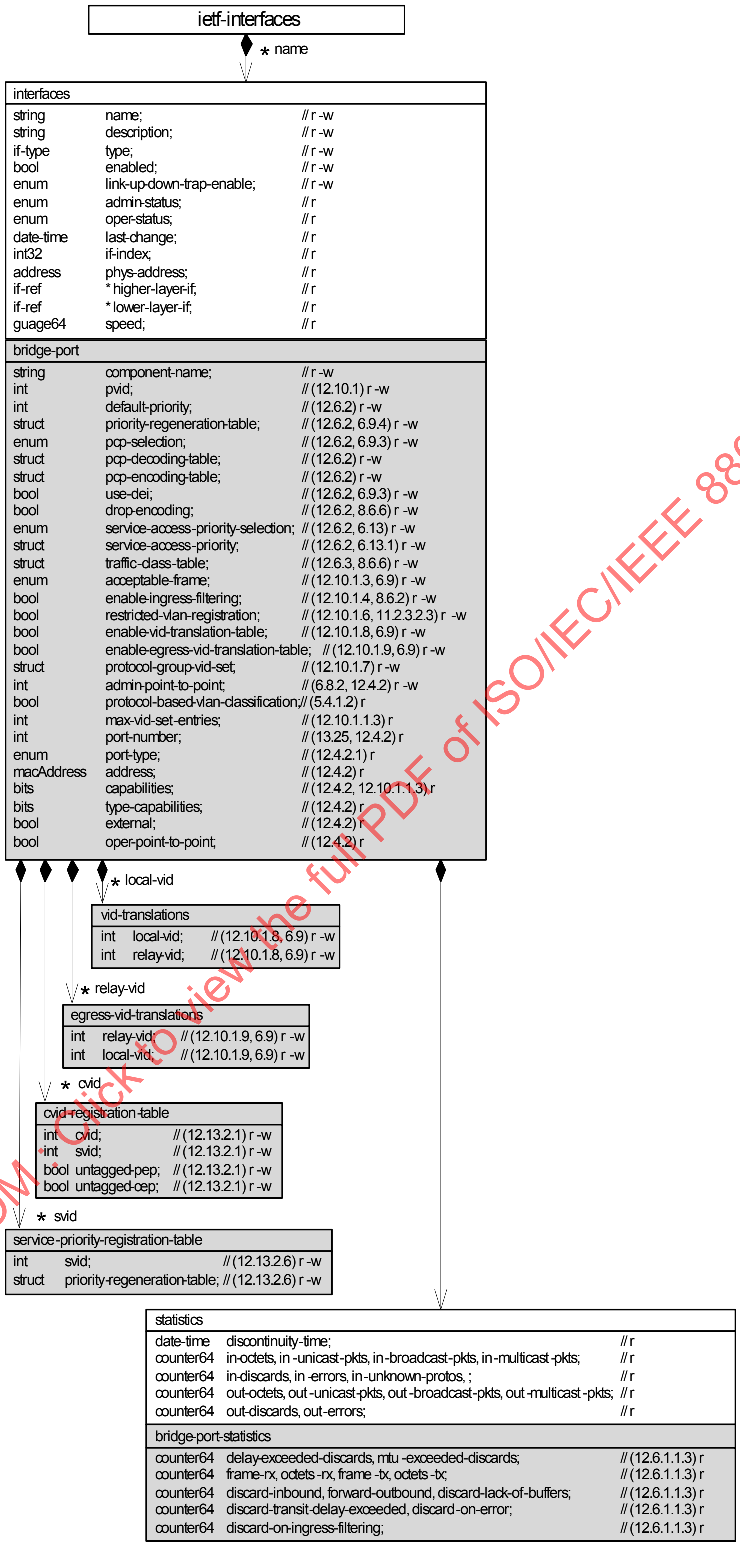

Provider bridge
string macAddress name; // (12.4) r-w
macAddress address; // (12.4) r-w
enum type; // r-w
int ports; // (12.4) r
counter32 up-time; // (12.4) r
int components; // r
* name (0..M)
* name (1)
C-VLAN component
string name; // r-w
int id; // (12.3) r-w
enum type; // (12.3) r-w
macAddress address; // (8.13.8, 13.24) r-w
bool traffic-class-enabled; // (12.4.1.5.1) r-w
int ports; // (12.4.1.1.3) r
if-ref *bridge-port // r
struct capabilities // (12.4.1.5.2) r
S-VLAN component
string name; // r-w
int id; // (12.3) r-w
enum type; // (12.3) r-w
macAddress address; // (8.13.8, 13.24) r-w
bool traffic-class-enabled; // (12.4.1.5.1) r-w
int ports; // (12.4.1.1.3) r
if-ref *bridge-port // r
struct capabilities // (12.4.1.5.2) r
filtering-database
permanent-database
bridge-vlan
bridge-mst
filtering-database
permanent-database
bridge-vlan
bridge-mst
filtering-database
int aging-time; // (12.7, 8.8.3) r-w
int size; // (12.7) r
int static-entries; // (12.7, 8.8.1) r
int dynamic-entries; // (12.7, 8.8.1) r
int static-vlan-registration-entries; // (12.7, 8.8.2) r
int dynamic-vlan-registration-entries; // (12.7, 8.8.5) r
int mac-address-registration-entries; // (12.7, 8.8.4) r
* database-id, address, vid
filtering-entries
int database-id; // (12.7.7) r-w
macAddress address; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum entry-type; // (12.7.7) r-w
enum status; // r
* port-number
port-map
if-ref port-number; // (8.8.1) r-w
enum map; // (8.8.1) r-w
* database-id, vid
vlan-registration-entries
int database-id; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum entry-type; // (12.7.7) r-w
* port-number
port-map
if-ref port-number; // (8.8.1) r-w
struct map; // (8.8.1) r-w
permanent-database
int size; // (12.7.6) r
int static-entries; // (12.7.6) r
int static-vlan-registration-entries; // (12.7.6) r
* database-id, address, vid
filtering-entries
int database-id; // (12.7.7) r-w
macAddress address; // (12.7.7) r-w
int vid; // (12.7.7) r-w
enum status; // r
* port-number
port-map
if-ref port-number; // (8.8.1) r-w
enum map; // (8.8.1) r-w
bridge-vlan
int version; // (12.10.1.3) r
int max-vids; // (12.10.1.3) r
bool override-default-pvid; // (12.10.1.3) r
stuct protocol-template; // (12.10.1.7) r
int max-msti; // (12.10.1.7) r
* vid
vlan-id
int vid; // (12.10.2) r-w
string name; // (12.10.2) r-w
if-ref *untagged-ports; // (8.8.2, 12.10.2.1.3) r
if-ref *egress-ports; // (8.8.10, 12.10.2.1.3) r
* vids
vid-to-fid-allocation
int *vids; // (12.10.3) r-w
int fid; // (12.10.3) r-w
enum allocation-type; // (12.10.3) r
protocol-group-database
string frame-format-type; // (12.10.1.7) r-w
struct frame-format; // (12.10.1.7) r-w
int protocol-group-id; // (6.12.2) r-w

Figure 48-8—Provider Bridge model

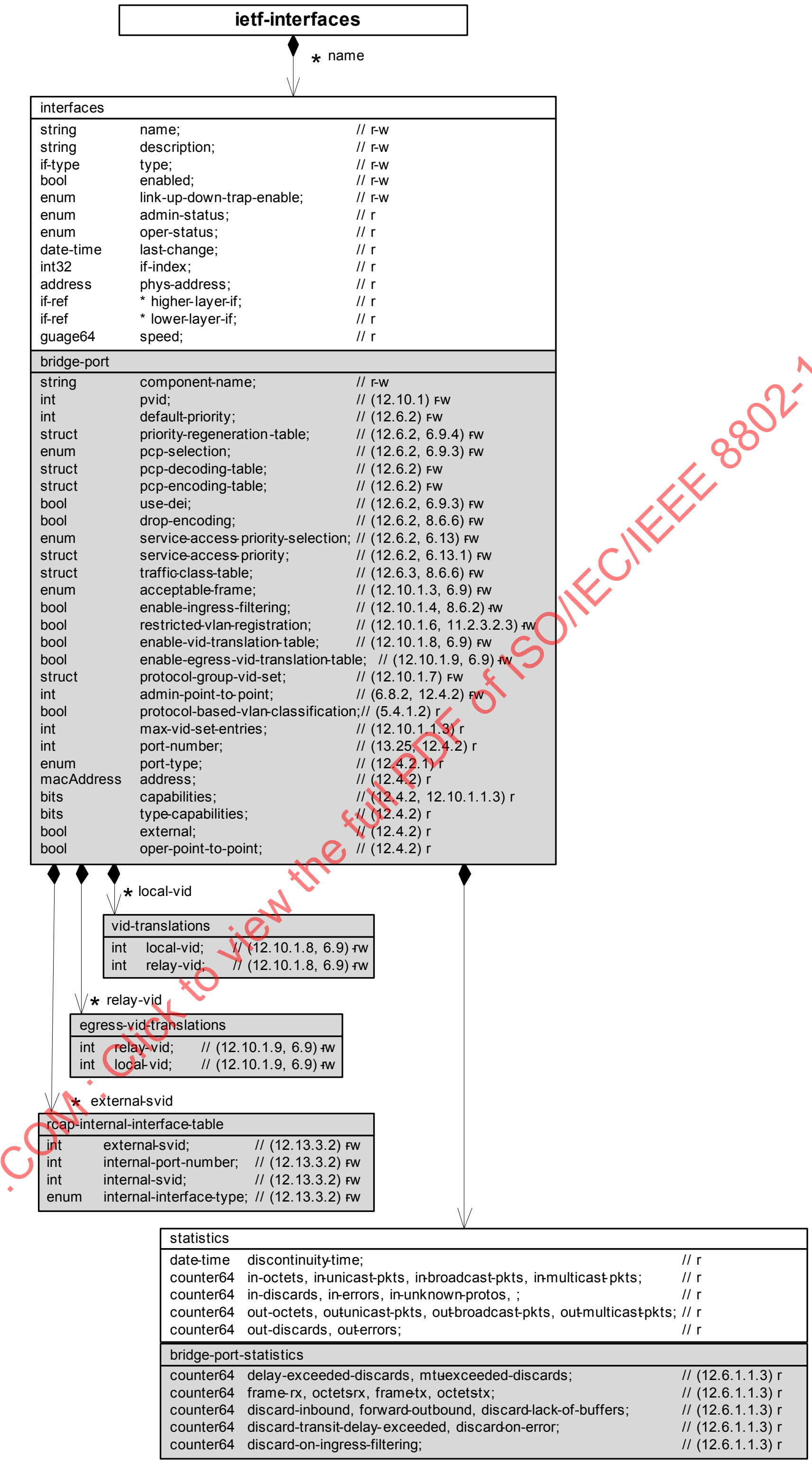
The Provider Edge Bridge C-VLAN component port (Interface) UML model is illustrated in Figure 48-9.


ietf-interfaces
* name
interfaces
string name; // r -w
string description; // r -w
if-type type; // r -w
bool enabled; // r -w
enum link-up-down-trap-enable; // r -w
enum admin-status; // r
enum oper-status; // r
date-time last-change; // r
int32 if-index; // r
address phys-address; // r
if-ref * higher-layer-if; // r
if-ref * lower-layer-if; // r
guage64 speed; // r
bridge-port
string component-name; // r -w
int pvid; // (12.10.1) r -w
int default-priority; // (12.6.2) r -w
struct priority-regeneration-table; // (12.6.2, 6.9.4) r -w
enum pcp-selection; // (12.6.2, 6.9.3) r -w
struct pcp-decoding-table; // (12.6.2) r -w
struct pcp-encoding-table; // (12.6.2) r -w
bool use-dei; // (12.6.2, 6.9.3) r -w
bool drop-encoding; // (12.6.2, 8.6.6) r -w
enum service-access-priority-selection; // (12.6.2, 6.13) r -w
struct service-access-priority; // (12.6.2, 6.13.1) r -w
struct traffic-class-table; // (12.6.3, 8.6.6) r -w
enum acceptable-frame; // (12.10.1.3, 6.9) r -w
bool enable-ingress-filtering; // (12.10.1.4, 8.6.2) r -w
bool restricted-vlan-registration; // (12.10.1.6, 11.2.3.2.3) r -w
bool enable-vid-translation-table; // (12.10.1.8, 6.9) r -w
bool enable-egress-vid-translation-table; // (12.10.1.9, 6.9) r -w
struct protocol-group-vid-set; // (12.10.1.7) r -w
int admin-point-to-point; // (6.8.2, 12.4.2) r -w
bool protocol-based-vlan-classification;// (5.4.1.2) r
int max-vid-set-entries; // (12.10.1.1.3) r
int port-number; // (13.25, 12.4.2) r
enum port-type; // (12.4.2.1) r
macAddress address; // (12.4.2) r
bits capabilities; // (12.4.2, 12.10.1.1.3) r
bits type-capabilities; // (12.4.2) r
bool external; // (12.4.2) r
bool oper-point-to-point; // (12.4.2) r
* local-vid
vid-translations
int local-vid; // (12.10.1.8, 6.9) r -w
int relay-vid; // (12.10.1.8, 6.9) r -w
* relay-vid
egress-vid-translations
int relay-vid; // (12.10.1.9, 6.9) r -w
int local-vid; // (12.10.1.9, 6.9) r -w
* cvid
cvid-registration-table
int cvid; // (12.13.2.1) r -w
int svid; // (12.13.2.1) r -w
bool untagged-pep; // (12.13.2.1) r -w
bool untagged-cep; // (12.13.2.1) r -w
* svid
service-priority-registration-table
int svid; // (12.13.2.6) r -w
struct priority-regeneration-table; // (12.13.2.6) r -w
statistics
date-time discontinuity-time; // r
counter64 in-octets, in -unicast-pkts, in-broadcast-pkts, in-multicast -pkts; // r
counter64 in-discards, in -errors, in-unknown-protos, ; // r
counter64 out-octets, out -unicast-pkts, out -broadcast-pkts, out -multicast -pkts; // r
counter64 out-discards, out-errors; // r
bridge-port-statistics
counter64 delay-exceeded-discards, mtu -exceeded-discards; // (12.6.1.1.3) r
counter64 frame-rx, octets -rx, frame -tx, octets -tx; // (12.6.1.1.3) r
counter64 discard-inbound, forward-outbound, discard-lack-of-buffers; // (12.6.1.1.3) r
counter64 discard-transit-delay-exceeded, discard-on-error; // (12.6.1.1.3) r
counter64 discard-on-ingress-filtering; // (12.6.1.1.3) r

NOTE—Items shaded in gray indicate extensions provided by the IEEE 802.1Q YANG models.

Figure 48-9—Provider Edge Bridge C-VLAN Interface mode

The Provider Bridge S-VLAN bridge component port (interface) UML model is shown in Figure 48-10.


ietf-interfaces
* name
interfaces
string name; // r-w
string description; // r-w
if-type type; // r-w
bool enabled; // r-w
enum link-up-down-trap-enable; // r-w
enum admin-status; // r
enum oper-status; // r
date-time last-change; // r
int32 if-index; // r
address phys-address; // r
if-ref * higher-layer-if; // r
if-ref * lower-layer-if; // r
guage64 speed; // r
bridge-port
string component-name; // r-w
int pvid; // (12.10.1) r-w
int default-priority; // (12.6.2) r-w
struct priority-regeneration-table; // (12.6.2, 6.9.4) r-w
enum pcp-selection; // (12.6.2, 6.9.3) r-w
struct pcp-decoding-table; // (12.6.2) r-w
struct pcp-encoding-table; // (12.6.2) r-w
bool use-dei; // (12.6.2, 6.9.3) r-w
bool drop-encoding; // (12.6.2, 8.6.6) r-w
enum service-access-priority-selection; // (12.6.2, 6.13) r-w
struct service-access-priority; // (12.6.2, 6.13.1) r-w
struct traffic-class-table; // (12.6.3, 8.6.6) r-w
enum acceptable-frame; // (12.10.1.3, 6.9) r-w
bool enable-ingress-filtering; // (12.10.1.4, 8.6.2) r-w
bool restricted-vlan-registration; // (12.10.1.6, 11.2.3.2.3) r-w
bool enable-vid-translation-table; // (12.10.1.8, 6.9) r-w
bool enable-egress-vid-translation-table; // (12.10.1.9, 6.9) r-w
struct protocol-group-vid-set; // (12.10.1.7) r-w
int admin-point-to-point; // (6.8.2, 12.4.2) r-w
bool protocol-based-vlan-classification;// (5.4.1.2) r
int max-vid-set-entries; // (12.10.1.1.3) r
int port-number; // (13.25, 12.4.2) r
enum port-type; // (12.4.2.1) r
macAddress address; // (12.4.2) r
bits capabilities; // (12.4.2, 12.10.1.1.3) r
bits type-capabilities; // (12.4.2) r
bool external; // (12.4.2) r
bool oper-point-to-point; // (12.4.2) r
* local-vid
vid-translations
int local-vid; // (12.10.1.8, 6.9) r-w
int relay-vid; // (12.10.1.8, 6.9) r-w
* relay-vid
egress-vid-translations
int relay-vid; // (12.10.1.9, 6.9) r-w
int local-vid; // (12.10.1.9, 6.9) r-w
* external-svid
rcap-internal-interface-table
int external-svid; // (12.13.3.2) r-w
int internal-port-number; // (12.13.3.2) r-w
int internal-svid; // (12.13.3.2) r-w
enum internal-interface-type; // (12.13.3.2) r-w
statistics
date-time discontinuity-time; // r
counter64 in-octets, in-unicast-pkts, in-broadcast-pkts, in-multicast-pkts; // r
counter64 in-discards, in-errors, in-unknown-protos, ; // r
counter64 out-octets, out-unicast-pkts, out-broadcast-pkts, out-multicast-pkts; // r
counter64 out-discards, out-errors; // r
bridge-port-statistics
counter64 delay-exceeded-discards, mtu-exceeded-discards; // (12.6.1.1.3) r
counter64 frame-rx, octets-rx, frame-tx, octets-tx; // (12.6.1.1.3) r
counter64 discard-inbound, forward-outbound, discard-lack-of-buffers; // (12.6.1.1.3) r
counter64 discard-transit-delay-exceeded, discard-on-error; // (12.6.1.1.3) r
counter64 discard-on-ingress-filtering; // (12.6.1.1.3) r

NOTE—Items shaded in gray indicate extensions provided by the IEEE 802.1Q YANG models.

Figure 48-10—Provider Edge Bridge S-VLAN interface model

48.4 Structure of the YANG model

IEEE 802.1Q YANG models are divided into a number of YANG modules. A summary of the modules contained in this clause is represented in Table 48-1.

Table 48-1—Structure of the YANG modules

Module	References	Notes
ieee802-types	48.6.3.1	General type definitions used within IEEE 802 standards.
ieee802-dot1q-types	48.6.3.2	General type definitions used by IEEE 802.1Q standard.
ieee802-dot1q-bridge	48.6.3.3	Generic IEEE 802.1Q Bridge YANG model, which is augmented by specific IEEE 802.1Q bridges.
ieee802-dot1q-tpmr	48.6.3.4	Two-Port MAC Relay YANG model, which augments the generic bridge YANG module.
ieee802-dot1q-vlan-bridge	48.6.3.5	Customer VLAN Bridge YANG model, which augments the generic bridge YANG module.
ieee802-dot1q-pb	48.6.3.6	Provider Bridges YANG model, which augments the generic bridge YANG module.

48.4.1 Structure of ieee802-dot1q-bridge YANG module

The *ieee802-dot1q-bridge* YANG module is divided into a number of YANG branches (e.g., subtrees). A summary of the YANG subtrees associated with this module is presented in Table 48-2.

Table 48-2—Structure of the ieee802-dot1q-bridge module

Branches	References	Notes
Bridge	48.5.1	—
Component	48.5.2	Bridge component (e.g., I-component, B-component, C-VLAN component, S-VLAN component, VLAN unaware component, EVC station edge relay component) within a system.
Filtering DB	48.5.3	Contains filtering information used by the Forwarding Process (8.6) in deciding through which ports of the Bridge frame should be forwarded.
Permanent DB	48.5.4	Permanent database object.
Bridge VLAN	48.5.5	Bridge component VLAN resources.
Bridge MST	48.5.6	Bridge MST resources.
Bridge Port	48.5.7	—
VID Translations	48.5.7	Set of relay VID to local VID bindings.
Egress VID Translations	48.5.7	Set of egress relay VID to local VID bindings.
Statistics	48.5.7	Statistics relating to the performance and traffic analysis.

The Bridge structure is illustrated in Figure 48-4. The Bridge port configurable and operational data model is illustrated in Figure 48-5.

48.4.2 Structure of ieee802-dot1q-tpmr YANG module

The *ieee802-dot1q-tpmr* YANG module is divided into a number of YANG branches (i.e., subtrees). A summary of the YANG subtrees associated with this module is presented in Table 48-3.

Table 48-3—Structure of the ieee802-dot1q-tpmr module

Branches	References	Notes
TPMR	48.4.2	Augments the generic ieee802-dot1q-bridge module only when the number of components is 1 and the bridge type is two-port-mac-relay-bridge.
Bridge Port	48.4.2	Augments the generic ieee802-dot1q-bridge module bridge-port branch, only when the number of bridge ports is 2.
MAC Status Propagation	48.4.2	—

The TPMR structure is illustrated in Figure 48-6. The TPMR port configurable and operational data model is illustrated in Figure 48-7.

48.4.3 Structure of ieee802-dot1q-vlan-bridge YANG module

The *ieee802-dot1q-vlan-bridge* YANG module is divided into a number of YANG branches (i.e., subtrees). A summary of the YANG subtrees associated with this module is presented in Table 48-4.

Table 48-4—Structure of the ieee802-dot1q-vlan-bridge module

Branches	References	Notes
VLAN Bridge	48.4.3	Augments the generic ieee802-dot1q-bridge module only when the number of components is 1 and the bridge type is customer-vlan-bridge.
Bridge Port	48.4.3	Augments the generic ieee802-dot1q-bridge module bridge-port branch.

The Customer VLAN Bridge structure is illustrated in Figure 48-4. The structure of the Customer VLAN Bridge Port configurable and operational status data models are illustrated in Figure 48-5.

48.4.4 Structure of ieee802-dot1q-pb YANG module

The *ieee802-dot1q-pb* YANG module is divided into a number of YANG branches (i.e., subtrees). A summary of the YANG subtrees associated with this module is presented in Table 48-5.

Table 48-5—Structure of the ieee802-dot1q-pb module

Branches	References	Notes
Provider Bridge	48.4.4	Utilizes the generic ieee802-dot1q-bridge module.
Bridge Port	48.4.4	Augments the generic ieee802-dot1q-bridge module bridge-port branch when the component type is cVlanComponent and the bridge port type is customer-edge-port.

The Provider Bridge structure is illustrated in Figure 48-8. The Provider Bridge port configurable and operational data model is illustrated in Figure 48-9 and Figure 48-10.

48.5 Relationship to IEEE 802.1Q managed objects

This standard specifies a Unified Modeling Language (UML) [B78] information model and a YANG data model that allows configuration and status reporting for bridges and bridge components including Media Access Control (MAC) Bridges, Two-Port MAC Relays (TPMRs), Customer Virtual Local Area Network (VLAN) Bridges, and Provider Bridges (as specified by this standard) with the capabilities currently specified in 12.4 to 12.8, 12.10, 12.13, and 12.19 of this standard.

In support of this standard, the YANG data model extends the IETF Interface Management YANG model (as specified in IETF RFC 8343).

The Bridge Port YANG node augments the Interface Management YANG model. The specific Bridge (e.g., TPMR, Customer VLAN, Provider Bridge) YANG models are augmentations from the Bridge YANG model. A system implementing these YANG models shall implement the *ieee802-dot1q-bridge*, *ieee802-types*, and *ieee802-dot1q-types* YANG models.

A system implementing the TPMR YANG model shall implement the *ieee802-dot1q-tpmr* YANG model. A system implementing the Customer VLAN Bridge YANG model shall implement the *ieee802-dot1q-vlan-bridge* YANG model. A system implementing the Provider Bridges YANG model shall implement the *ieee802-dot1q-pb* YANG model.

48.5.1 Relationship of the IEEE 802.1Q YANG bridge node

Table 48-6—Bridge cross-reference table

Generic bridge management information (Figure 48-4)	YANG node(s)
*** Bridge**	**ieee802-dot1q-bridge:bridges:bridge**
name (12.4) r-w	name — KEY
address (12.4) r-w	address
type r-w	bridge-type
ports (12.4) r	ports
upTime (12.4) r	up-time
components (12.3)	components

48.5.2 Relationship of the IEEE 802.1Q YANG component node

Table 48-7—Bridge component cross-reference table

Generic bridge management information (Figure 48-4)	**YANG node(s)**
*** Component**	**ieee802-dot1q-bridge:bridges:bridge:component**
—	name — KEY
id (12.3) r-w	id
type (12.3) r-w	type
address (8.13.8, 13.24)	address
trafficClassEnabled (12.4.1.5.1) r-w	traffic-class-enabled
ports (12.4.1.1.3) r	ports
* bridgePorts (—) r	* bridge-ports
Capabilities	**ieee802-dot1q-bridge:bridges:bridge:component:capabilities**
extendedFiltering (12.4.1.5.2) r	extended-filtering
trafficClasses (12.4.1.5.2) r	traffic-classes
staticEntryIndividualPort (12.4.1.5.2) r	static-entry-individual-port
ivlCapable (12.4.1.5.2) r	ivl-capable
svlCapable (12.4.1.5.2) r	svl-capable
hybridCapable (12.4.1.5.2) r	hybrid-capable
configurablePvidTagging (12.4.1.5.2) r	configurable-pvid-tagging
localVlanCapable (12.4.1.5.2) r	local-vlan-capable

48.5.3 Relationship of the IEEE 802.1Q YANG filtering database node

Table 48-8—Bridge component filtering database cross-reference table

Generic bridge management information (Figure 48-4)	YANG node(s)
Filtering Database	**ieee802-dot1q-bridge:bridges:bridge:component:filtering-database**
agingTime (12.7, 8.8.3) r-w	aging-time
size (12.7) r	size
staticEntries (12.7, 8.8.1) r	static-entries
dynamicEntries (12.7, 8.8.3) r	dynamic-entries
staticVlanRegistrationEntries (12.7, 8.8.2) r	static-vlan-registration-entries
dynamicVlanRegistrationEntries (12.7, 8.8.5) r	dynamic-vlan-registration-entries
macAddressRegistrationEntries (12.7, 8.8.4) r	mac-address-registration-entries
*** Filtering Entries**	**ieee802-dot1q-bridge:bridges:bridge:component:filtering-database:filtering-entries**
databaseId (12.7.7) r-w	database-id — KEY
address (12.7.7) r-w	address — KEY
vid (12.7.7) r-w	vid — KEY
entryType (12.7.7) r-w	entry-type
portMap (8.8.1, 8.8.2) r-w	port-map
status (—) r	status
*** VLAN Registration Entries**	**ieee802-dot1q-bridge:bridges:bridge:component:filtering-database:vlan-registration-entries**
databaseId (12.7.7) r-w	database-id — KEY
vid (12.7.7) r-w	vid — KEY
entryType (12.7.7) r-w	entry-type
portMap (8.8.1, 8.8.2) r-w	port-map

48.5.4 Relationship of the IEEE 802.1Q YANG permanent database node

Table 48-9—Bridge component permanent database cross-reference table

Generic bridge management information (Figure 48-4)	YANG node(s)
Permanent Database	**ieee802-dot1q-bridge:bridges:bridge:component:permanent-database**
size (12.7.6) r	size
staticEntries (12.7.6) r	static-entries
staticVlanRegistrationEntries (12.7.6) r	static-vlan-registration-entries
*** Filtering Entries**	**ieee802-dot1q-bridge:bridges:bridge:component:permanent-database:filtering-entries**
databaseId (12.7.7) r-w	database-id — KEY
address (12.7.7) r-w	address — KEY
vid (12.7.7) r-w	vid — KEY
portMap (8.8.1, 8.8.2) r-w	port-map

48.5.5 Relationship of the IEEE 802.1Q YANG bridge VLAN node

Table 48-10—Bridge component Bridge VLAN cross-reference table

Generic bridge management information (Figure 48-4)	YANG node(s)
Bridge VLAN	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan**
version (12.10.1.3) r	version
maxVids (12.10.1.3) r	max-vids
overrideDefaultPvid (12.10.1.3) r	override-default-pvid
protocolTemplate (12.10.1.7) r	protocol-template
maxMsti (12.10.1.7) r	max-msti
*** VLAN ID**	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan:vlan-id**
vid (12.10.2) r-w	vid — KEY
name (12.10.2) r-w	name
vid (12.10.2)	vid
* untaggedPorts (8.8.2, 12.10.2.1.3) r	* untagged-ports
* egressPorts (8.8.10, 12.10.2.1.3) r	* egress-ports
Protocol Group Database	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan:protocol-group-database**
frameFormatType (12.10.1.7) r-w	frame-format-type
protocolGroupId (6.12.2) r-w	protocol-group-id

Table 48-10—Bridge component Bridge VLAN cross-reference table *(continued)*

Generic bridge management information (Figure 48-4)	YANG node(s)
VID to FID	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan:vid-to-fid**
vid (12.10.3.4) r-w	vid
fid (12.10.3.4) r-w	fid
VID to FID Allocation	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan:vid-to-fid-allocation**
vid (12.10.3.2) r-w	vid — KEY
fid (12.10.3).2 r	fid
allocationType (12.10.3.2) r	allocation-type
FID to VID Allocation	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-vlan:fid-to-vid-allocation**
fid (12.10.3.3) r-w	fid — KEY
* vid (12.10.3) r	* vid
* allocationType (12.10.3) r	* allocation-type

48.5.6 Relationship of the IEEE 802.1Q YANG bridge MST node

Table 48-11—Bridge component Bridge MST cross-reference table

Generic bridge management information (Figure 48-4)	YANG node(s)
Bridge MST	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-mst**
*** MSTID (12.12.1)**	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-mst:mstid**
FID to MSTID	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-mst:fid-to-mstid**
fid (12.12.2) r-w	fid — KEY
mstid (12.12.2) r-w	mstid
FID to MSTID Allocation	**ieee802-dot1q-bridge:bridges:bridge:component:bridge-mst:fid-to-mstid-allocation**
fids (12.12.2) r-w	fids — KEY
mstid (12.12.2) r	mstid

48.5.7 Relationship of the IEEE 802.1Q YANG bridge port node

A system implementing this YANG model also implements the Interface Management YANG model defined in IETF RFC 8343. The Interface Management YANG data model defines the management of network interfaces.

NOTE— Since network interfaces are central to the management of many Internet protocols, it is important to establish a common data model for how interfaces are identified, configured, and monitored.

The IEEE 802.1Q Bridge Port nodes augments the Interface Management YANG model.

Table 48-12—Bridge port cross-reference table

Generic bridge management information (Figure 48-5)	YANG node(s)
Bridge Port	**ietf-interfaces:interfaces:interface:ieee802-dot1q-bridge:bridge-port**
componentName r-w	component-name
pvid (5.4, 12.10.1) r-w	pvid
defaultPriority (12.6.2) r-w	default-priority
priorityRegenerationTable (12.6.2, 6.9.4) r-w	priority-regeneration-table
pcpSelection (12.6.2, 6.9.3) r-w	pcp-selection
pcpDecodingTable (12.6.2, 6.9.3) r-w	pcp-decoding-table
pcpEncodingTable (12.6.2) r-w	pcp-encoding-table
useDei (12.6.2, 6.9.3) r-w	use-dei
dropEncoding (12.6.2, 8.6.6) r-w	drop-encoding
serviceAccessPrioritySelection (12.6.2, 6.13) r-w	service-access-priority-selection
serviceAccessPriority (12.6.2, 6.13.1) r-w	service-access-priority
trafficClass (112.6.3, 8.6.6) r-w	traffic-class
acceptableFrame (12.10.1.3, 6.9) r-w	acceptable-frame
enableIngressFiltering (12.10.1.4, 8.6.2) r-w	enable-ingress-filtering
restrictedVlanRegistration (12.10.1.6, 11.2.3.2.3) r-w	enable-restricted-vlan-registration
vidTranslationTable (12.10.1.8, 6.9) r-w	enable-vid-translation-table
egressVidTranslationTable (12.10.1.9, 6.9) r-w	enable-egress-vid-translation-table
protocolGroupId (6.12.2) r-w	protocol-group-id
protocolGroupDatabaseContents (12.10.1.7) r-w	protocol-group-vid-set
adminPointToPoint (6.8.2, 12.4.2) r-w	admin-point-to-point
* vidTranslations (12.10.1.8, 6.9) r-w	* vid-translations
* egressVidTranslations (12.10.1.9, 6.9) r-w	* egress-vid-translations
protocolBasedVlanClassification (5.4.1.2) r	protocol-based-vlan-classification
maxVidSetEntries (12.10.1.1.3) r	max-vid-set-entries
portNumber (13.25, 12.4.2) r	port-number
portType (12.4.2.1) r	port-type
address (12.4.2) r	address
capabilities (12.4.2, 12.10.1.1.3) r	capabilities
typeCapabilities (12.4.2) r	type-capabilities
external (12.4.2) r	external
operPointToPoint (12.4.2) r	oper-point-to-point

Table 48-12—Bridge port cross-reference table *(continued)*

Generic bridge management information (Figure 48-5)	YANG node(s)
Bridge Port Statistics	
delayExceededDiscard (12.6.1.1.3, 8.6.6) r	delay-exceeded-discard
mtuExceededDiscards (12.6.1.1.3) r	mtu-exceeded-discards
frameRx (12.6.1.1.3) r	frame-rx
octetsRx (12.6.1.1.3) r	octets-rx
frameTx () r	frame-tx
octetsTx () r	octets-tx
discardInbound (12.6.1.1.3) r	discard-inbound
forwardOutbound (12.6.1.1.3) r	forward-outbound
discardLackOfBuffers (12.6.1.1.3) r	discard-lack-of-buffers
discardTransitDelayExceeded (12.6.1.1.3) r	discard-transit-delay-exceeded
discardOnError (12.6.1.1.3) r	discard-on-error
discardOnIngressFiltering (12.6.1.1.3) r	discard-on-ingress-filtering

48.6 Definition of IEEE 802.1Q YANG modules

The structure of the IEEE 802.1Q related YANG modules is described in 48.4. In the following YANG module definitions, if any discrepancy between the DESCRIPTION text and the corresponding definition in any other part of this standard occur, the definitions outside this subclause take precedence.

48.6.1 YANG data scheme tree definitions

A simplified graphical representation of the data model is used in this document. The meaning of the symbols in these diagrams is as follows:

- Brackets “[” and “]” enclose list keys.
- Abbreviations before data node names: “rw” means configuration (read-write), and “ro” means state data (read-only).
- Symbols after data node names: “?” means an optional node, “!” means a presence container, and “*” denotes a list and leaf-list.
- Parentheses enclose choice and case nodes, and case nodes are also marked with a colon (“:”).
- Ellipsis (“...”) stands for contents of subtrees that are not shown.

48.6.2 YANG data scheme definition for ieee802-dot1q-bridge YANG module

```
module: ieee802-dot1q-bridge
    +--rw bridges
       +--rw bridge* [name]
          +--rw name            dot1qtypes:name-type
          +--rw address         ieee:mac-address
          +--rw bridge-type     identityref
          +--ro ports?          uint16
          +--ro up-time?        yang:zero-based-counter32
          +--ro components?     uint32
          +--rw component* [name]
             +--rw name                     string
             +--rw id?                      uint32
             +--rw type                     identityref
             +--rw address?                 ieee:mac-address
             +--rw traffic-class-enabled?   boolean
             +--ro ports?                   uint16
             +--ro bridge-port*             if:interface-ref
             +--ro capabilities
             |  +--ro extended-filtering?             boolean
             |  +--ro traffic-classes?                boolean
             |  +--ro static-entry-individual-port?   boolean
             |  +--ro ivl-capable?                    boolean
             |  +--ro svl-capable?                    boolean
             |  +--ro hybrid-capable?                 boolean
             |  +--ro configurable-pvid-tagging?      boolean
             |  +--ro local-vlan-capable?             boolean
             +--rw filtering-database
             |  +--rw aging-time?                          uint32
             |  +--ro size?                                yang:gauge32
             |  +--ro static-entries?                      yang:gauge32
             |  +--ro dynamic-entries?                     yang:gauge32
             |  +--ro static-vlan-registration-entries?    yang:gauge32
             |  +--ro dynamic-vlan-registration-entries?   yang:gauge32
            |  +--ro mac-address-registration-entries?    yang:gauge32 {extended-
filtering-services}?
             |  +--rw filtering-entry* [database-id vids address]
             |  |  +--rw database-id    uint32
             |  |  +--rw address        ieee:mac-address
             |  |  +--rw vids           dot1qtypes:vid-range-type
             |  |  +--rw entry-type?    enumeration
             |  |  +--rw port-map* [port-ref]
             |  |  |  +--rw port-ref                            port-number-type
             |  |  |  +--rw (map-type)?
             |  |  |     +--:(static-filtering-entries)
             |  |  |     |  +--rw static-filtering-entries
             |  |  |     |     +--rw control-element?         enumeration
             |  |  |     |     +--rw connection-identifier?   port-number-type
             |  |  |     +--:(static-vlan-registration-entries)
             |  |  |     |  +--rw static-vlan-registration-entries
             |  |  |     |     +--rw registrar-admin-control?   enumeration
             |  |  |     |     +--rw vlan-transmitted?          enumeration
             |  |  |     +--:(mac-address-registration-entries)
             |  |  |     |  +--rw mac-address-registration-entries
             |  |  |     |     +--rw control-element?   enumeration
             |  |  |     +--:(dynamic-vlan-registration-entries)
             |  |  |     |  +--rw dynamic-vlan-registration-entries
             |  |  |     |     +--rw control-element?   enumeration
             |  |  |     +--:(dynamic-reservation-entries)
             |  |  |     |  +--rw dynamic-reservation-entries
             |  |  |     |     +--rw control-element?   enumeration
             |  |  |     +--:(dynamic-filtering-entries)
             |  |  |        +--rw dynamic-filtering-entries
             |  |  |           +--rw control-element?   enumeration
             |  |  +--ro status?        enumeration
             |  +--rw vlan-registration-entry* [database-id vids]
             |     +--rw database-id    uint32
             |     +--rw vids           dot1qtypes:vid-range-type
             |     +--rw entry-type?    enumeration
             |     +--rw port-map* [port-ref]
             |        +--rw port-ref                            port-number-type
             |        +--rw (map-type)?
             |           +--:(static-filtering-entries)
             |           |  +--rw static-filtering-entries
             |           |     +--rw control-element?         enumeration
             |           |     +--rw connection-identifier?   port-number-type
             |           +--:(static-vlan-registration-entries)
             |           |  +--rw static-vlan-registration-entries
             |           |     +--rw registrar-admin-control?   enumeration
             |           |     +--rw vlan-transmitted?          enumeration
             |           +--:(mac-address-registration-entries)
             |           |  +--rw mac-address-registration-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-vlan-registration-entries)
             |           |  +--rw dynamic-vlan-registration-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-reservation-entries)
             |           |  +--rw dynamic-reservation-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-filtering-entries)
             |              +--rw dynamic-filtering-entries
             |                 +--rw control-element?   enumeration
             +--rw permanent-database
             |  +--ro size?                               yang:gauge32
             |  +--ro static-entries?                     yang:gauge32
             |  +--ro static-vlan-registration-entries?   yang:gauge32
             |  +--rw filtering-entry* [database-id vids address]
             |     +--rw database-id    uint32
             |     +--rw address        ieee:mac-address
             |     +--rw vids           dot1qtypes:vid-range-type
             |     +--ro status?        enumeration
             |     +--rw port-map* [port-ref]
             |        +--rw port-ref                            port-number-type
             |        +--rw (map-type)?
             |           +--:(static-filtering-entries)
             |           |  +--rw static-filtering-entries
             |           |     +--rw control-element?         enumeration
             |           |     +--rw connection-identifier?   port-number-type
             |           +--:(static-vlan-registration-entries)
             |           |  +--rw static-vlan-registration-entries
             |           |     +--rw registrar-admin-control?   enumeration
             |           |     +--rw vlan-transmitted?          enumeration
             |           +--:(mac-address-registration-entries)
             |           |  +--rw mac-address-registration-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-vlan-registration-entries)
             |           |  +--rw dynamic-vlan-registration-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-reservation-entries)
             |           |  +--rw dynamic-reservation-entries
             |           |     +--rw control-element?   enumeration
             |           +--:(dynamic-filtering-entries)
             |              +--rw dynamic-filtering-entries
             |                 +--rw control-element?   enumeration
             +--rw bridge-vlan
             |  +--ro version?                  uint16
             |  +--ro max-vids?                 uint16
             |  +--ro override-default-pvid?     boolean
            |  +--ro protocol-template?        dot1qtypes:protocol-frame-format-
type {port-and-protocol-based-vlan}?
             |  +--ro max-msti?                 uint16
             |  +--rw vlan* [vid]
             |  |  +--rw vid               dot1qtypes:vlan-index-type
             |  |  +--rw name?             dot1qtypes:name-type
             |  |  +--ro untagged-ports*   if:interface-ref
             |  |  +--ro egress-ports*     if:interface-ref
             |  +--rw protocol-group-database* [db-index] {port-and-protocol-
based-vlan}?
             |  |  +--rw db-index             uint16
             |  |  +--rw frame-format-type?   dot1qtypes:protocol-frame-format-
type
             |  |  +--rw (frame-format)?
             |  |  |  +--:(ethernet-rfc1042-snap8021H)
             |  |  |  |  +--rw ethertype?           dot1qtypes:ethertype-type
             |  |  |  +--:(snap-other)
             |  |  |  |  +--rw protocol-id?         string
             |  |  |  +--:(llc-other)
             |  |  |     +--rw dsap-ssap-pairs
             |  |  |        +--rw llc-address?   string
             |  |  +--rw group-id?            uint32
             |  +--rw vid-to-fid-allocation* [vids]
             |  |  +--rw vids               dot1qtypes:vid-range-type
             |  |  +--ro fid?               uint32
             |  |  +--ro allocation-type?   enumeration
             |  +--rw fid-to-vid-allocation* [fid]
             |  |  +--rw fid                uint32
             |  |  +--ro allocation-type?   enumeration
             |  |  +--ro vid*               dot1qtypes:vlan-index-type
             |  +--rw vid-to-fid* [vid]
             |     +--rw vid    dot1qtypes:vlan-index-type
             |     +--rw fid?   uint32
             +--rw bridge-mst
                +--rw mstid*                     dot1qtypes:mstid-type
                +--rw fid-to-mstid* [fid]
                |  +--rw fid      uint32
                |  +--rw mstid?   dot1qtypes:mstid-type
                +--rw fid-to-mstid-allocation* [fids]
                   +--rw fids      dot1qtypes:vid-range-type
                   +--rw mstid?   dot1qtypes:mstid-type
  augment /if:interfaces/if:interface:
    +--rw bridge-port
       +--rw component-name?                        string
       +--rw port-type?                             identityref
       +--rw pvid?                                  dot1qtypes:vlan-index-type
       +--rw default-priority?                      dot1qtypes:priority-type
       +--rw priority-regeneration
       |  +--rw priority0?   priority-type
       |  +--rw priority1?   priority-type
       |  +--rw priority2?   priority-type
       |  +--rw priority3?   priority-type
       |  +--rw priority4?   priority-type
       |  +--rw priority5?   priority-type
       |  +--rw priority6?   priority-type
       |  +--rw priority7?   priority-type
       +--rw pcp-selection?                        dot1qtypes:pcp-selection-type
       +--rw pcp-decoding-table
       |  +--rw pcp-decoding-map* [pcp]
       |     +--rw pcp             pcp-selection-type
       |     +--rw priority-map* [priority-code-point]
       |        +--rw priority-code-point    priority-type
       |        +--rw priority?              priority-type
       |        +--rw drop-eligible?         boolean
       +--rw pcp-encoding-table
       |  +--rw pcp-encoding-map* [pcp]
       |     +--rw pcp             pcp-selection-type
       |     +--rw priority-map* [priority dei]
       |        +--rw priority               priority-type
       |        +--rw dei                    boolean
       |        +--rw priority-code-point?   priority-type
       +--rw use-dei?                               boolean
       +--rw drop-encoding?                         boolean
       +--rw service-access-priority-selection?     boolean
       +--rw service-access-priority
       |  +--rw priority0?   priority-type
       |  +--rw priority1?   priority-type
       |  +--rw priority2?   priority-type
       |  +--rw priority3?   priority-type
       |  +--rw priority4?   priority-type
       |  +--rw priority5?   priority-type
       |  +--rw priority6?   priority-type
       |  +--rw priority7?   priority-type
       +--rw traffic-class
       |  +--rw traffic-class-map* [priority]
       |     +--rw priority                   priority-type
       |     +--rw available-traffic-class* [num-traffic-class]
       |        +--rw num-traffic-class    uint8
       |        +--rw traffic-class?       traffic-class-type
       +--rw acceptable-frame?                      enumeration
       +--rw enable-ingress-filtering?              boolean
       +--rw enable-restricted-vlan-registration?   boolean
       +--rw enable-vid-translation-table?          boolean
       +--rw enable-egress-vid-translation-table?   boolean
       +--rw protocol-group-vid-set* [group-id] {port-and-protocol-based-vlan}?
       |  +--rw group-id    uint32
       |  +--rw vid*        dot1qtypes:vlanid
       +--rw admin-point-to-point?                  enumeration
       +--ro protocol-based-vlan-classification?    boolean {port-and-protocol-
based-vlan}?
       +--ro max-vid-set-entries?                   uint16 {port-and-protocol-
based-vlan}?
       +--ro port-number?                           dot1qtypes:port-number-type
       +--ro address?                               ieee:mac-address
       +--ro capabilities?                          bits
       +--ro type-capabilties?                      bits
       +--ro external?                              boolean
       +--ro oper-point-to-point?                   boolean
       +--ro statistics
       |  +--ro delay-exceeded-discards?          yang:counter64
       |  +--ro mtu-exceeded-discards?            yang:counter64
       |  +--ro frame-rx?                         yang:counter64
       |  +--ro octets-rx?                        yang:counter64
       |  +--ro frame-tx?                         yang:counter64
       |  +--ro octets-tx?                        yang:counter64
       |  +--ro discard-inbound?                  yang:counter64
       |  +--ro forward-outbound?                 yang:counter64
       |  +--ro discard-lack-of-buffers?          yang:counter64
       |  +--ro discard-transit-delay-exceeded?   yang:counter64
       |  +--ro discard-on-error?                 yang:counter64
       |  +--ro discard-on-ingress-filtering?     yang:counter64 {ingress-
filtering}?
       +--rw vid-translations* [local-vid]
       |  +--rw local-vid    dot1qtypes:vlanid
       |  +--rw relay-vid?   dot1qtypes:vlanid
       +--rw egress-vid-translations* [relay-vid]
          +--rw relay-vid    dot1qtypes:vlanid
          +--rw local-vid?   dot1qtypes:vlanid
```

48.6.2.1 YANG data scheme definition for ieee802-dot1q-tpmr YANG module

```
module: ieee802-dot1q-tpmr
  augment /if:interfaces/if:interface/dot1q:bridge-port:
    +--rw managed-address?          boolean
    +--rw mac-status-propagation
       +--rw link-notify?         boolean
       +--rw link-notify-wait?    yang:timeticks
       +--rw link-notify-retry?   yang:timeticks
       +--rw mac-notify?          boolean
       +--rw mac-notify-time?     yang:timeticks
       +--rw mac-recover-time?    yang:timeticks
  augment /if:interfaces/if:interface/dot1q:bridge-port/dot1q:statistics:
    +--ro acks-tx?                    yang:counter64
    +--ro add-notificatons-tx?        yang:counter64
    +--ro loss-notification-tx?       yang:counter64
    +--ro loss-confirmation-tx?       yang:counter64
    +--ro acks-rx?                    yang:counter64
    +--ro add-notificatons-rx?        yang:counter64
    +--ro loss-notification-rx?       yang:counter64
    +--ro loss-confirmation-rx?       yang:counter64
    +--ro add-events?                 yang:counter64
    +--ro loss-events?                yang:counter64
    +--ro mac-status-notifications?   yang:counter64
```

48.6.2.2 YANG data scheme definition for ieee802-dot1q-pb YANG module

```
module: ieee802-dot1q-pb
  augment /if:interfaces/if:interface/dot1q:bridge-port:
    +--rw svid?                            dot1qtypes:vlanid
    +--rw cvid-registration* [cvid]
    |  +--rw cvid            dot1qtypes:vlanid
    |  +--rw svid?           dot1qtypes:vlanid
    |  +--rw untagged-pep?   boolean
    |  +--rw untagged-cep?   boolean
    +--rw service-priority-regeneration* [svid]
    |  +--rw svid                     dot1qtypes:vlanid
    |  +--rw priority-regeneration
    |     +--rw priority0?   priority-type
    |     +--rw priority1?   priority-type
    |     +--rw priority2?   priority-type
    |     +--rw priority3?   priority-type
    |     +--rw priority4?   priority-type
    |     +--rw priority5?   priority-type
    |     +--rw priority6?   priority-type
    |     +--rw priority7?   priority-type
    +--rw rcap-internal-interface* [external-svid]
       +--rw external-svid              dot1qtypes:vlanid
       +--rw internal-port-number?      dot1qtypes:port-number-type
       +--rw internal-svid?             dot1qtypes:vlanid
       +--rw internal-interface-type?   enumeration
```

48.6.3 YANG data module definitions

48.6.3.1 Definition for the ieee802-types YANG module

```
module ieee802-types {
  namespace urn:ieee:std:802.1Q:yang:ieee802-types;
  prefix ieee;
  organization
    "IEEE 802.1 Working Group";
  contact
   "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
            P.O. Box 1331
            Piscataway
            NJ 08854
            USA

    E-mail: STDS-802-1-L@IEEE.ORG";
  description
    "This module contains a collection of generally useful derived
    data types for IEEE YANG models.";
  revision 2018-03-07 {
    description
      "Published as part of IEEE Std 802.1Q-2018.
      Initial version.";
    reference
      "IEEE Std 802.1Q-2018, Bridges and Bridged Networks.";
  }

  typedef mac-address {
    type string {
      pattern "[0-9a-fA-F]{2}(-[0-9a-fA-F]{2}){5}";
    }
    description
      "The mac-address type represents a MAC address in the canonical
      format and hexadecimal format specified by IEEE Std 802. The
      hexidecimal representation uses uppercase characters.";
    reference
      "3.1 of IEEE Std 802-2014
      8.1 of IEEE Std 802-2014";
  }
}
```

48.6.3.2 Definition for the ieee802-dot1q-types YANG module

```
module ieee802-dot1q-types {
  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://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
            P.O. Box 1331
            Piscataway
            NJ 08854
            USA

    E-mail: STDS-802-1-L@IEEE.ORG";
  description
    "Common types used within dot1Q-bridge modules.";
  revision 2018-03-07 {
    description
      "Published as part of IEEE Std 802.1Q-2018.
      Initial version.";
    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
      from.";
  }
  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-2018";
  }
  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-2018";
  }
  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..65535";
    }
    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 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-2018";
}
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-2018";
}
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 of IEEE Std 802.1Q-2018
      6.9.3 of IEEE Std 802.1Q-2018";
  }
  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-2018";
  }
  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 802-2014";
  }
  typedef dot1q-tag-type {
    type identityref {
      base dot1q-vlan-type;
    }
    description
      "Identifies a specific 802.1Q tag type";
    reference
      "IEEE Std 802.1Q-2018";
  }
  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
      "3.239 of IEEE Std 802.1Q-2018";
  }
  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-2018";
    leaf priority0 {
      type priority-type;
      default "0";
      description
        "Priority 0";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority1 {
      type priority-type;
      default "1";
      description
        "Priority 1";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority2 {
      type priority-type;
      default "2";
      description
        "Priority 2";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority3 {
      type priority-type;
      default "3";
      description
        "Priority 3";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority4 {
      type priority-type;
      default "4";
      description
        "Priority 4";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority5 {
      type priority-type;
      default "5";
      description
        "Priority 5";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority6 {
      type priority-type;
      default "6";
      description
        "Priority 6";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
    leaf priority7 {
      type priority-type;
      default "7";
      description
        "Priority 7";
      reference
        "12.6.2.3 of IEEE Std 802.1Q-2018
        6.9.4 of IEEE Std 802.1Q-2018";
    }
  }
  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-2018";
    list pcp-decoding-map {
      key "pcp";
      description
        "This map associates the priority code point field found in
        the VLAN to 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 of IEEE Std 802.1Q-2018
          6.9.3 of IEEE Std 802.1Q-2018";
      }
      list priority-map {
        key "priority-code-point";
        description
          "This map associated a priority code point value to priority
          and drop eligible parameters.";
        leaf priority-code-point {
          type priority-type;
          description
            "Priority associated with the pcp.";
          reference
            "12.6.2.7 of IEEE Std 802.1Q-2018
            6.9.3 of IEEE Std 802.1Q-2018";
        }
        leaf priority {
          type priority-type;
          description
            "Priority associated with the pcp.";
          reference
            "12.6.2.7 of IEEE Std 802.1Q-2018
            6.9.3 of IEEE Std 802.1Q-2018";
        }
        leaf drop-eligible {
          type boolean;
          description
            "Drop eligible value for pcp";
          reference
            "12.6.2.7 of IEEE Std 802.1Q-2018
            6.9.3 of IEEE Std 802.1Q-2018";
        }
      }
    }
  }
  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 of IEEE Std 802.1Q-2018
      6.9.3 of IEEE Std 802.1Q-2018";
    list pcp-encoding-map {
      key "pcp";
      description
        "This map associated the priority and drop-eligible parameters
        to 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 of IEEE Std 802.1Q-2018
          6.9.3 of IEEE Std 802.1Q-2018";
      }
      list priority-map {
        key "priority dei";
        description
          "This map associated the priority and drop-eligible
          parameters to 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 of IEEE Std 802.1Q-2018
            6.9.3 of IEEE Std 802.1Q-2018";
        }
        leaf dei {
          type boolean;
          description
            "The drop eligible value.";
          reference
            "12.6.2 of IEEE Std 802.1Q-2018
            8.6.6 of IEEE Std 802.1Q-2018";
        }
        leaf priority-code-point {
          type priority-type;
          description
            "PCP value for priority when DEI value";
          reference
            "12.6.2.9 of IEEE Std 802.1Q-2018
            6.9.3 of IEEE Std 802.1Q-2018";
        }
      }
    }
  }
  grouping service-access-priority-table-grouping {
    description
      "The Service Access Priority Table associates a received
      priority with a serice access priority.";
    reference
      "12.6.2.17 of IEEE Std 802.1Q-2018
      6.13.1 of IEEE Std 802.1Q-2018";
    leaf priority0 {
      type priority-type;
      default "0";
      description
        "Service access priority value for priority 0";
      reference
        "12.6.2.17 of IEEE Std 802.1Q-2018
        6.13.1 of IEEE Std 802.1Q-2018";
    }
    leaf priority1 {
      type priority-type;
      default "1";
      description
        "Service access priority value for priority 1";
      reference
```

```
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority2 {
        type priority-type;
        default "2";
        description
          "Service access priority value for priority 2";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority3 {
        type priority-type;
        default "3";
        description
          "Service access priority value for priority 3";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority4 {
        type priority-type;
        default "4";
        description
          "Service access priority value for priority 4";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority5 {
        type priority-type;
        default "5";
        description
          "Service access priority value for priority 5";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority6 {
        type priority-type;
        default "6";
        description
          "Service access priority value for priority 6";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
      leaf priority7 {
        type priority-type;
        default "7";
        description
          "Service access priority value for priority 7";
        reference
          "12.6.2.17 of IEEE Std 802.1Q-2018
          6.13.1 of IEEE Std 802.1Q-2018";
      }
    }
    grouping traffic-class-table-grouping {
      description
        "The Traffic Class Table models the operations that can be
        performed on, or inquire about, the current contents of the
        Traffic Class Table (8.6.6) for a given Port.";
      reference
        "12.6.3 of IEEE Std 802.1Q-2018
        8.6.6 of IEEE Std 802.1Q-2018";
      list traffic-class-map {
        key "priority";
        description
          "The priority index into the 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-2018";
        }
        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-2018";
          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-2018";
          }
          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-2018";
          }
        }
      }
    }
    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 of IEEE Std 802.1Q-2018
        8.8.2 of IEEE Std 802.1Q-2018";
      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-2018";
        }
        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
                "containing 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-2018";
            }
            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 of IEEE Std 802.1Q-2018
                8.8.12 of IEEE Std 802.1Q-2018";
            }
          }
          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-2018";
            }
            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-2018";
            }
          }
          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
                "containing 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-2018";
            }
          }
          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
                "containing 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-2018";
            }
          }
          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
                "Containing 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-2018";
            }
          }
          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
                "Containing 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-2018";
            }
          }
        }
      }
    }
    grouping bridge-port-statistics-grouping {
      description
        "Grouping of bridge port statistics.";
      reference
        "12.6.1.1.3 of IEEE Std 802.1Q-2018";
      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 of IEEE Std 802.1Q-2018
          8.6.6 of IEEE Std 802.1Q-2018";
      }
      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
          "12.6.1.1.3, item g) of IEEE Std 802.1Q-2018";
      }
      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-2018";
      }
      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-2018";
      }
      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-2018";
      }
      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-2018";
      }
      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-2018";
      }
      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-2018";
      }
      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-2018";
      }
    }
  }
```

48.6.3.3 Definition for the ieee802-dot1q-bridge YANG module

```
module ieee802-dot1q-bridge {
  namespace urn:ieee:std:802.1Q:yang:ieee802-dot1q-bridge;
  prefix dot1q;
  import ieee802-types {
    prefix ieee;
  }
  import ietf-yang-types {
    prefix yang;
  }
  import ietf-interfaces {
    prefix if;
  }
  import iana-if-type {
    prefix ianaif;
  }
  import ieee802-dot1q-types {
    prefix dot1qtypes;
  }
  organization
    "IEEE 802.1 Working Group";
  contact
    "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
            P.O. Box 1331
            Piscataway
            NJ 08854
            USA

    E-mail: STDS-802-1-L@IEEE.ORG";
  description
    "This YANG module describes the bridge configuration model for the
    following IEEE 802.1Q Bridges:
    1) Two Port MAC Relays
    2) Customer VLAN Bridges
    3) Provider Bridges.";
  revision 2018-03-07 {
    description
      "Published as part of IEEE Std 802.1Q-2018.
      Initial version.";
    reference
      "IEEE Std 802.1Q-2018, Bridges and Bridged Networks.";
  }

  feature ingress-filtering {
    description
      "Each Port may support an Enable Ingress Filtering parameter. A
      frame received on a Port that is not in the member set (8.8.10)
      associated with the frames VID shall be discarded if this
      parameter is set. The default value for this parameter is reset,
      i.e., Disable Ingress Filtering, for all Ports. Any Port that
      supports setting this parameter shall also support resetting it.
      The parameter may be configured by the management operations
      defined in Clause 12.";
    reference
      "8.6.2 of IEEE Std 802.1Q-2018";
  }
  feature extended-filtering-services {
    description
      "Extended Filtering Services support the filtering behavior
      required for regions of a network in which potential recipients
      of multicast frames exist, and where both the potential
      recipients of frames and the Bridges are able to support dynamic
      configuration of filtering information for group MAC addresses.
      In order to integrate this extended filtering behavior with the
      needs of regions of the network that support only Basic
      Filtering Services, Bridges that support Extended Filtering
      Services can be statically and dynamically configured to modify
      their filtering behavior on a per-group MAC address basis, and
      also on the basis of the overall filtering service provided by
      each outbound Port with regard to multicast frames. The latter
      capability permits configuration of the Ports default forwarding
      or filtering behavior with regard to group MAC addresses for
      which no specific static or dynamic filtering information has
      been configured.";
    reference
      "8.8.4 of IEEE Std 802.1Q-2018
      Clause 10 of IEEE Std 802.1Q-2018";
  }
  feature port-and-protocol-based-vlan {
    description
      "A VLAN-aware bridge component implementation in conformance to
      the provisions of this standard for Port-and-Protocol-based VLAN
```

```
      classification (5.4.1) shall 1) Support one or more of the
      following Protocol Classifications and Protocol Template
      formats: Ethernet, RFC_1042, SNAP_8021H, SNAP_Other, or
      LLC_Other (6.12); and may 2) Support configuration of the
      contents of the Protocol Group Database.";
    reference
      "5.4.1.2 of IEEE Std 802.1Q-2018";
  }
  feature flow-filtering {
    description
      "Flow filtering support enables Bridges to distinguish frames
      belonging to different client flows and to use this information
      in the forwarding process. Information related to client flows
      may be used at the boundary of an SPT Domain to generate a flow
      hash value. The flow hash, carried in an F-TAG, serves to
      distinguish frames belonging to different flows and can be used
      in the forwarding process to distribute frames over equal cost
      paths. This provides for finer granularity load spreading while
      maintaining frame order for each client flow.";
    reference
      "44.2 of IEEE Std 802.1Q-2018";
  }
  feature simple-bridge-port {
    description
      "A simple bridge port allows underlying (MAC) layers to share
      the same Interface as the Bridge Port.";
  }
  feature flexible-bridge-port {
    description
      "A flexible bridge port supports an Interface that is a Bridge
      Port to be a separate Interface from the underlying (MAC) layer.";
  }

  identity type-of-bridge {
    description
      "Represents the configured Bridge type.";
  }
  identity customer-vlan-bridge {
    base type-of-bridge;
    description
      "Base identity for a Customer VLAN Bridge.";
  }
  identity provider-bridge {
    base type-of-bridge;
    description
      "Base identity for a Provider Bridge (PB).";
  }
  identity provider-edge-bridge {
    base type-of-bridge;
    description
      "Base identity for a Provider Edge Bridge (PEB).";
  }
  identity two-port-mac-relay-bridge {
    base type-of-bridge;
    description
      "Base identity for a Two Port MAC Relay (TPMR).";
  }
  identity type-of-component {
    description
      "Represents the type of Component.";
  }
  identity c-vlan-component {
    base type-of-component;
    description
      "Base identity for a C-VLAN component.";
  }
  identity s-vlan-component {
    base type-of-component;
    description
      "Base identity for a S-VLAN component.";
  }
  identity d-bridge-component {
    base type-of-component;
    description
      "Base identity for a VLAN unaware component.";
  }
  identity edge-relay-component {
    base type-of-component;
    description
      "Base identity for an EVB station ER component.";
  }
  identity type-of-port {
    description
      "Represents the type of Bridge port.";
  }
  identity c-vlan-bridge-port {
    base type-of-port;
    description
      "Indicates the port can be a C-TAG aware port of an enterprise
      VLAN aware Bridge.";
  }
  identity provider-network-port {
    base type-of-port;
    description
      "Indicates the port can be an S-TAG aware port of a Provider
      Bridge or Backbone Edge Bridge used for connections within a PBN
      (Provider Bridged Network) or PBBN (Provider Backbone Bridged
      Network).";
  }
  identity customer-network-port {
    base type-of-port;
    description
      "Indicates the port can be an S-TAG aware port of a Provider
      Bridge or Backbone Edge Bridge used for connections to the
      exterior of a PBN (Provider Bridged Network) or PBBN (Provider
      Backbone Bridged Network).";
  }
  identity customer-edge-port {
    base type-of-port;
    description
      "Indicates the port can be a C-TAG aware port of a Provider
      Bridge used for connections to the exterior of a PBN (Provider
      Bridged Network) or PBBN (Provider Backbone Bridged Network).";
  }
  identity d-bridge-port {
    base type-of-port;
    description
      "Indicates the port can be a VLAN-unaware member of an 802.1Q
      Bridge.";
  }
  identity remote-customer-access-port {
    base type-of-port;
    description
      "Indicates the port can be an S-TAG aware port of a Provider
      Bridge capable of providing Remote Customer Service Interfaces.";
  }
  identity bridge-interface {
    description
      "Generic interface property that represents any interface that
      can be associated with an IEEE 802.1Q compliant Bridge
      component. Any new Interface types would derive from this
      identity to automatically pick up Bridge related configuration
      or operational data.";
  }

  container bridges {
    description
      "Contains the Bridge(s) configuration information.";
    list bridge {
      key "name";
      unique "address";
      description
        "Provides configuration data in support of the Bridge
        Configuration resources. There is a single bridge data node
        per Bridge.";
      leaf name {
        type dot1qtypes:name-type;
        description
          "A text string associated with the Bridge, of locally
          determined significance.";
        reference
          "12.4 of IEEE Std 802.1Q-2018";
      }
      leaf address {
        type ieee:mac-address;
        mandatory true;
        description
          "The MAC address for the Bridge from which the Bridge
          Identifiers used by the STP, RSTP, and MSTP are derived.";
        reference
          "12.4 of IEEE Std 802.1Q-2018";
      }
      leaf bridge-type {
        type identityref {
          base type-of-bridge;
        }
        mandatory true;
        description
          "The type of Bridge.";
      }
      leaf ports {
        type uint16 {
          range "1..4095";
        }
        config false;
        description
          "The number of Bridge Ports (MAC Entities)";
        reference
          "12.4 of IEEE Std 802.1Q-2018";
      }
      leaf up-time {
        type yang:zero-based-counter32;
        units "seconds";
        config false;
        description
          "The count in seconds of the time elapsed since the Bridge
          was last reset or initialized.";
        reference
          "12.4 of IEEE Std 802.1Q-2018";
      }
      leaf components {
        type uint32;
        config false;
        description
          "The number of components associated with the Bridge.";
      }
      list component {
        key "name";
        description
          "The set of components associated with a given Bridge. For
          example, - A TPMR is associated with a single VLAN
          unaware component. - A Customer VLAN Bridge is associated
          with a single VLAN aware component. - A Provider Bridge is
          associated with a single S-VLAN component and zero or more
          C-VLAN components.";
        reference
          "12.3 of IEEE Std 802.1Q-2018";
        leaf name {
          type string;
          description
            "The name of the Component.";
        }
        leaf id {
          type uint32;
          description
            "Unique identifier for a particular Bridge component
```

```
            within the system.";
          reference
            "12.3, item l) of IEEE Std 802.1Q-2018";
        }
        leaf type {
          type identityref {
            base type-of-component;
          }
          mandatory true;
          description
            "The type of component used to classify a particular
            Bridge component within a Bridge system comprising
            multiple components.";
          reference
            "12.3, item m) of IEEE Std 802.1Q-2018";
        }
        leaf address {
          type ieee:mac-address;
          description
            "Unique EUI-48 Universally Administered MAC address
            assigned to a Bridge component.";
          reference
            "13.24 of IEEE Std 802.1Q-2018
            8.13.8 of IEEE Std 802.1Q-2018";
        }
        leaf traffic-class-enabled {
          type boolean;
          default "true";
          description
            "Indication of Traffic Classes enablement associated with
            the Bridge Component. A value of True indicates that
            Traffic Classes are enabled on this Bridge Component. A
            value of False indicates that the Bridge Component
            operates with a single priority level for all traffic.";
          reference
            "12.4.1.5.1 of IEEE Std 802.1Q-2018";
        }
        leaf ports {
          type uint16 {
            range "1..4095";
          }
          config false;
          description
            "The number of Bridge Ports associated with the Bridge
            Component.";
          reference
            "12.4.1.1.3, item c) of IEEE Std 802.1Q-2018";
        }
        leaf-list bridge-port {
          type if:interface-ref;
          config false;
          description
            "List of bridge-port references.";
        }
        container capabilities {
          config false;
          description
            "Array of Boolean values of the feature capabilities
            associated with a given Bridge Component.";
          reference
            "12.10.1.1.3, item b) of IEEE Std 802.1Q-2018
            12.4.1.5.2 of IEEE Std 802.1Q-2018";
          leaf extended-filtering {
            type boolean;
            default "false";
            description
              "Can perform filtering on individual multicast addresses
              controlled by MMRP.";
            reference
              "12.4.1.5.2 of IEEE Std 802.1Q-2018";
          }
          leaf traffic-classes {
```

```
        type boolean;
        default "false";
        description
          "Can map priority to multiple traffic classes.";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf static-entry-individual-port {
        type boolean;
        default "false";
        description
          "Static entries per port.";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf ivl-capable {
        type boolean;
        default "true";
        description
          "Independent VLAN Learning (IVL).";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf svl-capable {
        type boolean;
        default "false";
        description
          "Shared VLAN Learning (SVL).";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf hybrid-capable {
        type boolean;
        default "false";
        description
          "Both IVL and SVL simultaneously.";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf configurable-pvid-tagging {
        type boolean;
        default "false";
        description
          "Whether the implementation supports the ability to
          override the default PVID setting and its egress status
          (VLAN-tagged or Untagged) on each port.";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
      leaf local-vlan-capable {
        type boolean;
        default "false";
        description
          "Can support multiple local Bridges, outside the scope
          of 802.1Q defined VLANs.";
        reference
          "12.4.1.5.2 of IEEE Std 802.1Q-2018";
      }
    }
    container filtering-database {
      when "../../bridge-type != 'two-port-mac-relay-bridge'" {
        description
          "Applies to non TPMRs.";
      }
      description
        "Contains filtering information used by the Forwarding
        Process in deciding through which Ports of the Bridge
        frames should be forwarded.";
      reference
        "12.7 of IEEE Std 802.1Q-2018";
      leaf aging-time {
        type uint32 {
```

```
    range "10..10000000";
  }
  units "seconds";
  default "300";
  description
    "The timeout period in seconds for aging out
    dynamically-learned forwarding information.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.3 of IEEE Std 802.1Q-2018";
}
leaf size {
  type yang:gauge32;
  config false;
  description
    "The maximum number of entries that can be held in the
    FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018";
}
leaf static-entries {
  type yang:gauge32;
  config false;
  description
    "The number of Static Filtering entries currently in the
    FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.1 of IEEE Std 802.1Q-2018";
}
leaf dynamic-entries {
  type yang:gauge32;
  config false;
  description
    "The number of Dynamic Filtering entries currently in
    the FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.3 of IEEE Std 802.1Q-2018";
}
leaf static-vlan-registration-entries {
  type yang:gauge32;
  config false;
  description
    "The number of Static VLAN Registration entries
    currently in the FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.2 of IEEE Std 802.1Q-2018";
}
leaf dynamic-vlan-registration-entries {
  type yang:gauge32;
  config false;
  description
    "The number of Dynamic VLAN Registration entries
    currently in the FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.5 of IEEE Std 802.1Q-2018";
}
leaf mac-address-registration-entries {
  if-feature "extended-filtering-services";
  type yang:gauge32;
  config false;
  description
    "The number of MAC Address Registration entries
    currently in the FDB.";
  reference
    "12.7 of IEEE Std 802.1Q-2018
    8.8.4 of IEEE Std 802.1Q-2018";
}
list filtering-entry {
```

```
key "database-id vids address";
description
  "Information for the entries associated with the
  Permanent Database.";
leaf database-id {
  type uint32;
  description
    "The identity of this Filtering Database.";
  reference
    "12.7.7 of IEEE Std 802.1Q-2018";
}
leaf address {
  type ieee:mac-address;
  description
    "A MAC address (unicast, multicast, broadcast) for
    which the device has forwarding and/or filtering
    information.";
  reference
    "12.7.7 of IEEE Std 802.1Q-2018";
}
leaf vids {
  type dot1qtypes:vid-range-type;
  description
    "The set of VLAN identifiers to which this entry
    applies.";
  reference
    "12.7.7 of IEEE Std 802.1Q-2018";
}
leaf entry-type {
  type enumeration {
    enum static {
      description
        "Static entry type";
    }
    enum dynamic {
      description
        "Dynamic/learnt entry type";
    }
  }
  description
    "The type of filtering entry. Whether static or
    dynamic. Static entries can be created, deleted, and
    retrieved. However, dynamic entries can only be
    deleted or retrieved by the management entity.
    Consequently, a Bridge is not required to accept a
    command that can alter the dynamic entries except
    delete a dynamic entry.";
  reference
    "12.7.7 of IEEE Std 802.1Q-2018";
}
uses dot1qtypes:port-map-grouping;
leaf status {
  type enumeration {
    enum other {
      description
        "None of the following. This may include the case
        where some other object is being used to determine
        if and how frames addressed to the value of the
        corresponding instance of 'address' are being
        forwarded.";
    }
    enum invalid {
      description
        "This entry is no longer valid (e.g., it was
        learned but has since aged out), but has not yet
        been flushed from the table.";
    }
    enum learned {
      description
        "The value of the corresponding instance of the
        port node was learned and is being used.";
    }
```

```
        enum self {
          description
            "The value of the corresponding instance of the
            address node representing one of the devices
            address.";
        }
        enum mgmt {
          description
            "The value of the corresponding instance of
            address node that is also the value of an existing
            instance.";
        }
      }
      config false;
      description
        "The status of this entry.";
    }
  }
  list vlan-registration-entry {
    key "database-id vids";
    description
      "The VLAN Registration Entries models the operations
      that can be performed on a single VLAN Registration
      Entry in the FDB. The set of VLAN Registration Entries
      within the FDB changes under management control and also
      as a result of MVRP exchanges";
    reference
      "12.7.5 of IEEE Std 802.1Q-2018";
    leaf database-id {
      type uint32;
      description
        "The identity of this Filtering Database.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    leaf vids {
      type dot1qtypes:vid-range-type;
      description
        "The set of VLAN identifiers to which this entry
        applies.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    leaf entry-type {
      type enumeration {
        enum static {
          description
            "Static entry type";
        }
        enum dynamic {
          description
            "Dynamic/learnt entry type";
        }
      }
      description
        "The type of filtering entry. Whether static or
        dynamic. Static entries can be created, deleted, and
        retrieved. However, dynamic entries can only be
        deleted or retrieved by the management entity.
        Consequently, a Bridge is not required to accept a
        command that can alter the dynamic entries except
        delete a dynamic entry.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    uses dot1qtypes:port-map-grouping;
  }
}
container permanent-database {
  description
    "The Permanent Database container models the operations
    that can be performed on, or affect, the Permanent
    Database. There is a single Permanent Database per FDB.";
  leaf size {
    type yang:gauge32;
    config false;
    description
      "The maximum number of entries that can be held in the
      FDB.";
    reference
      "12.7.6 of IEEE Std 802.1Q-2018";
  }
  leaf static-entries {
    type yang:gauge32;
    config false;
    description
      "The number of Static Filtering entries currently in the
      FDB.";
    reference
      "12.7.6 of IEEE Std 802.1Q-2018";
  }
  leaf static-vlan-registration-entries {
    type yang:gauge32;
    config false;
    description
      "The number of Static VLAN Registration entries
      currently in the FDB.";
    reference
      "12.7.6 of IEEE Std 802.1Q-2018";
  }
  list filtering-entry {
    key "database-id vids address";
    description
      "Information for the entries associated with the
      Permanent Database.";
    leaf database-id {
      type uint32;
      description
        "The identity of this Filtering Database.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    leaf address {
      type ieee:mac-address;
      description
        "A MAC address (unicast, multicast, broadcast) for
        which the device has forwarding and/or filtering
        information.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    leaf vids {
      type dot1qtypes:vid-range-type;
      description
        "The set of VLAN identifiers to which this entry
        applies.";
      reference
        "12.7.7 of IEEE Std 802.1Q-2018";
    }
    leaf status {
      type enumeration {
        enum other {
          description
            "None of the following. This may include the case
            where some other object is being used to determine
            if and how frames addressed to the value of the
            corresponding instance of 'address' are being
            forwarded.";
        }
        enum invalid {
          description
            "This entry is no longer valid (e.g., it was
            learned but has since aged out), but has not yet
            been flushed from the table.";
        }
        enum learned {
          description
            "The value of the corresponding instance of the
            port node was learned and is being used.";
        }
        enum self {
          description
            "The value of the corresponding instance of the
            address node representing one of the devices
            address.";
        }
        enum mgmt {
          description
            "The value of the corresponding instance of
            address node that is also the value of an existing
            instance.";
        }
      }
      config false;
      description
        "The status of this entry.";
    }
    uses dot1qtypes:port-map-grouping;
  }
}
container bridge-vlan {
  when "../../bridge-type != 'two-port-mac-relay-bridge'" {
    description
      "Applies to non TPMRs.";
  }
  description
    "The Bridge VLAN container models configuration
    information that modify, or inquire about, the overall
    configuration of the Bridges VLAN resources. There is a
    single Bridge VLAN Configuration managed object per
    Bridge.";
  reference
    "12.10 of IEEE Std 802.1Q-2018";
  leaf version {
    type uint16;
    config false;
    description
      "The version number supported.";
    reference
      "12.10.1.3 of IEEE Std 802.1Q-2018";
  }
  leaf max-vids {
    type uint16;
    config false;
    description
      "The maximum number of VIDs supported.";
    reference
      "12.10.1.3 of IEEE Std 802.1Q-2018";
  }
  leaf override-default-pvid {
    type boolean;
    default "false";
    config false;
    description
      "Indicates if the default PVID can be overridden, and
      its egress status (VLAN-tagged or untagged) on each
      port.";
    reference
      "12.10.1.3 of IEEE Std 802.1Q-2018";
  }
  leaf protocol-template {
    if-feature "port-and-protocol-based-vlan";
    type dot1qtypes:protocol-frame-format-type;
    config false;
    description
      "The data-link encapsulation format or the
      detagged_frame_type in a Protocol Template";
    reference
      "12.10.1.7 of IEEE Std 802.1Q-2018";
  }
  leaf max-msti {
    type uint16;
    config false;
    description
      "The maximum number of MSTIs supported within an MST
      region (i.e., the number of spanning tree instances that
      can be supported in addition to the CIST), for MST
      Bridges. For SST Bridges, this parameter may be either
      omitted or reported as 0.";
    reference
      "12.10.1.7 of IEEE Std 802.1Q-2018";
  }
  list vlan {
    key "vid";
    description
      "List of VLAN related configuration nodes associated
      with the Bridge.";
    reference
      "12.10.2 of IEEE Std 802.1Q-2018";
    leaf vid {
      type dot1qtypes:vlan-index-type;
      description
        "The VLAN identifier to which this entry applies.";
      reference
        "12.10.2 of IEEE Std 802.1Q-2018";
    }
    leaf name {
      type dot1qtypes:name-type;
      description
        "A text string of up to 32 characters of locally
        determined significance.";
      reference
        "12.10.2 of IEEE Std 802.1Q-2018";
    }
    leaf-list untagged-ports {
      type if:interface-ref;
      config false;
      description
        "The set of ports in the untagged set for this VID.";
      reference
        "12.10.2.1.3 of IEEE Std 802.1Q-2018
        8.8.2 of IEEE Std 802.1Q-2018";
    }
    leaf-list egress-ports {
      type if:interface-ref;
      config false;
      description
        "The set of egress ports in the member set for this
        VID.";
      reference
        "12.10.2.1.3 of IEEE Std 802.1Q-2018
        8.8.10 of IEEE Std 802.1Q-2018";
    }
  }
  list protocol-group-database {
    if-feature "port-and-protocol-based-vlan";
    key "db-index";
    description
      "List of the protocol group database entries.";
    reference
      "12.10.1.7 of IEEE Std 802.1Q-2018
      6.12.3 of IEEE Std 802.1Q-2018";
    leaf db-index {
      type uint16;
      description
        "The protocol group database index.";
    }
    leaf frame-format-type {
```

```
  type dot1qtypes:protocol-frame-format-type;
  description
    "The data-link encapsulation format or the
    detagged_frame_type in a Protocol Template";
  reference
    "12.10.1.7 of IEEE Std 802.1Q-2018";
}
choice frame-format {
  description
    "The identification of the protocol above the
    data-link layer in a Protocol Template. Depending on
    the frame type, the octet string will have one of the
    following values: - For ethernet, rfc1042 and
    snap8021H, this is the 16-bit (2-octet) IEEE 802
    Clause 9.3 EtherType field. - For snapOther, this is
    the 40-bit (5-octet) PID. - For llcOther, this is the
    2-octet IEEE 802.2 Link Service Access Point (LSAP)
    pair: first octet for Destination Service Access Point
    (DSAP) and second octet for Source Service Access
    Point (SSAP).";
  reference
    "12.10.1.7 of IEEE Std 802.1Q-2018";
  case ethernet-rfc1042-snap8021H {
    when
      "frame-format-type = 'Ethernet' or "+
      "frame-format-type = 'rfc1042' or "+
      "frame-format-type = 'snap8021H'" {
      description
        "Applies to Ethernet, RFC 1042, SNAP 8021H frame
        formats.";
    }
    description
      "Identifier used if Ethenet, RFC1042, or SNAP 8021H.";
    leaf ethertype {
      type dot1qtypes:ethertype-type;
      description
        "Format containing the 16-bit IEEE 802 EtherType
        field.";
      reference
        "9.3 of IEEE Std 802-2014";
    }
  }
  case snap-other {
    when "frame-format-type = 'snapOther'" {
      description
        "Applies to Snap Other frame formats.";
    }
    description
      "Identifier used if SNAP other.";
    leaf protocol-id {
      type string {
        pattern "[0-9a-fA-F]{2}(-[0-9a-fA-F]{2}){4}";
      }
      description
        "Format containing the 40-bit protocol identifier
        (PID). The canonical representation uses uppercase
        characters.";
      reference
        "12.10.1.7.1 of IEEE Std 802.1Q-2018";
    }
  }
  case llc-other {
    when "frame-format-type = 'llcOther'" {
      description
        "Applies to LLC Other frame formats";
    }
    description
      "Identifier used if LLC other.";
    container dsap-ssap-pairs {
      description
        "A pair of ISO/IEC 8802-2 DSAP and SSAP address
        field values, for matching frame formats of
```

```
          LLC_Other.";
        leaf llc-address {
          type string {
            pattern "[0-9a-fA-F]{2}-[0-9a-fA-F]{2}";
          }
          description
            "A pair of ISO/IEC 8802-2 DSAP and SSAP address
            field values, for matching frame formats of
            LLC_Other. The canonical representation uses
            uppercase characters.";
          reference
            "12.10.1.7.1 of IEEE Std 802.1Q-2018";
        }
      }
    }
  }
  leaf group-id {
    type uint32;
    description
      "Designates a group of protocols in the Protocol Group
      Database.";
    reference
      "6.12.2 of IEEE Std 802.1Q-2018";
  }
}
list vid-to-fid-allocation {
  key "vids";
  description
    "This list allows inquiries about VID to FID
    allocations.";
  leaf vids {
    type dot1qtypes:vid-range-type;
    description
      "Range of VLAN identifiers.";
    reference
      "12.10.3 of IEEE Std 802.1Q-2018";
  }
  leaf fid {
    type uint32;
    config false;
    description
      "The Filtering Database used by a set of VIDs.";
    reference
      "12.10.3 of IEEE Std 802.1Q-2018";
  }
  leaf allocation-type {
    type enumeration {
      enum undefined {
        description
          "No allocation defined.";
      }
      enum fixed {
        description
          "A fixed allocation to FID is defined.";
      }
      enum dynamic {
        description
          "A dynamic allocation to FID is defined.";
      }
    }
    config false;
    description
      "The type of allocation used";
    reference
      "12.10.3 of IEEE Std 802.1Q-2018";
  }
}
list fid-to-vid-allocation {
  key "fid";
  description
    "The FID to VID allocations managed object models
    operations that inquire about FID to VID allocations.";
```

```
        leaf fid {
          type uint32;
          description
            "The Filtering Database used by a set of VIDs.";
          reference
            "12.10.3 of IEEE Std 802.1Q-2018";
        }
        leaf allocation-type {
          type enumeration {
            enum undefined {
              description
                "No allocation defined.";
            }
            enum fixed {
              description
                "A fixed allocation to FID is defined.";
            }
            enum dynamic {
              description
                "A dynamic allocation to FID is defined.";
            }
          }
          config false;
          description
            "The type of allocation used";
          reference
            "12.10.3 of IEEE Std 802.1Q-2018";
        }
        leaf-list vid {
          type dot1qtypes:vlan-index-type;
          config false;
          description
            "The VLAN identifier to which this entry applies.";
          reference
            "12.7.7 of IEEE Std 802.1Q-2018";
        }
      }
      list vid-to-fid {
        key "vid";
        description
          "Fixed allocation of a VID to an FID. The underlying
          system will ensure that subsequent commands that make
          changes to the VID to FID mapping can override previous
          associations.";
        reference
          "12.10.3.4 of IEEE Std 802.1Q-2018
          12.10.3.5 of IEEE Std 802.1Q-2018";
        leaf vid {
          type dot1qtypes:vlan-index-type;
          description
            "A list of VLAN identifier associated with a given
            database identifier (i.e., FID).";
          reference
            "12.7.7 of IEEE Std 802.1Q-2018";
        }
        leaf fid {
          type uint32;
          description
            "The Filtering Database used by this VLAN";
          reference
            "12.10.3 of IEEE Std 802.1Q-2018";
        }
      }
    }
    container bridge-mst {
      when "../../bridge-type != 'two-port-mac-relay-bridge'" {
        description
          "Applies to non TPMRs.";
      }
      description
        "The Bridge MST container models configuration information
        that modify, or inquire about, the overall configuration
```